

VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Jnana Sangama, Belgaum-590018



A PROJECT REPORT (15CSP85) ON

“DISEASE PREDICTION USING MACHINE LEARNING”

Submitted in Partial fulfillment of the Requirements for the Degree of
Bachelor of Engineering in Computer Science & Engineering

By

MAYUR SHIVAKUMAR(1CR16CS087)

MALEEHA ARIF (1CR16CS079)

MEGHANA MENON (1CR16CS088)

Under the Guidance of,

MR. SHIVRAJ V.B

Assistant Professor, Dept. of CSE



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CMR INSTITUTE OF TECHNOLOGY

#132, AECS LAYOUT, IT PARK ROAD, KUNDALAHALLI, BANGALORE-560037

CMR INSTITUTE OF TECHNOLOGY

#132, AECS LAYOUT, IT PARK ROAD, KUNDALAHALLI, BANGALORE-560037

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



CERTIFICATE

Certified that the project work entitled “**DISEASE PREDICTION USING MACHINE LEARNING**” carried out by **Mr. MAYUR SHIVAKUMAR** USN 1CR16CS087, **Ms. MALEEHA ARIF**, USN 1CR16CS079, **Ms. MEGHANA MENON**, USN 1CR16CS088 , bonafide students of CMR Institute of Technology, in partial fulfillment for the award of **Bachelor of Engineering** in Computer Science and Engineering of the Visveswaraiah Technological University, Belgaum during the year 2019-2020. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the Report deposited in the departmental library.

The project report has been approved as it satisfies the academic requirements in respect of Project work prescribed for the said Degree.

Mr. Shivraj V.B
Assistant Professor
Dept. of CSE, CMRIT

Dr. Prem Kumar Ramesh
Professor & Head
Dept. of CSE, CMRIT

Dr. Sanjay Jain
Principal
CMRIT

External Viva

Name of the examiners

- 1.
- 2.

Signature with date

DECLARATION

We, the students of Computer Science and Engineering, CMR Institute of Technology, Bangalore declare that the work entitled "**DISEASE PREDICTION USING MACHINE LEARNING**" has been successfully completed under the guidance of Prof. Shivraj V.B, Computer Science and Engineering Department, CMR Institute of technology, Bangalore. This dissertation work is submitted in partial fulfillment of the requirements for the award of Degree of Bachelor of Engineering in Computer Science and Engineering during the academic year 2019 - 2020. Further the matter embodied in the project report has not been submitted previously by anybody for the award of any degree or diploma to any university.

Place: Bangalore

Date: 06/06/2020

Team members:

MAYUR SHIVAKUMAR (1CR16CS087)

MALEEHA ARIF (1CR16CS079)

MEGHANA MENON(1CR16CS088)

ABSTRACT

The project implements 3 linear models and one deep learning model: Naïve Bayes, Support Vector Machine, K-Nearest Neighbors network to investigate their performance on diabetes and heart disease datasets obtained from the UCI data repository. In addition to the comparison of the algorithms, each algorithm has been integrated into a prediction engine and exposed over an API. The project also includes a web platform to facilitate collaboration among researchers and doctors. As the results show, our prediction engine is capable of recognizing the presence of the disease and also predict it accurately. Performance improvements could also be achieved by using complex deep learning methods.

ACKNOWLEDGEMENT

We take this opportunity to express our sincere gratitude and respect to **CMR Institute of Technology, Bengaluru** for providing us a platform to pursue our studies and carry out our final year project

We have a great pleasure in expressing our deep sense of gratitude to **Dr. Sanjay Jain**, Principal, CMRIT, Bangalore, for his constant encouragement.

We would like to thank **Dr. Prem Kumar Ramesh**, Professor and Head, Department of Computer Science and Engineering, CMRIT, Bangalore, who has been a constant support and encouragement throughout the course of this project.

We consider it a privilege and honor to express our sincere gratitude to our guide **Mr. Shivraj V.B, Assistant Professor** , Department of Computer Science and Engineering, for the valuable guidance throughout the tenure of this review.

We also extend our thanks to all the faculty of Computer Science and Engineering who directly or indirectly encouraged us.

Finally, we would like to thank our parents and friends for all their moral support they have given us during the completion of this work.

TABLE OF CONTENTS

	Page No.
Certificate	ii
Declaration	iii
Abstract	iv
Acknowledgement	v
Table of contents	vi
List of Figures	viii
1 INTRODUCTION	1
1.1 Relevance of the Project	2
1.2 Problem Statement	3
1.3 Objectives	4
1.4 Scope of the Project	4
1.5 Project Methodology	5
2 LITERATURE SURVEY	8
3 SYSTEM REQUIREMENTS SPECIFICATION	15
3.1 Functional Requirements	15
3.2 Non-Functional Requirements	21
3.3 Hardware Requirements	22
3.4 Software Requirements	22
4 SYSTEM ANALYSIS AND DESIGN	24
4.1 Initial Engine Design	24
4.2 Use Cases	25
5 PROBLEM FORMULATION	27
5.1 Dataset Formulation	27
6 IMPLEMENTATION	30

6.1 Naïve Bayes	30
6.2 Support Vector Machine	30
6.3 K Nearest Neighbors	31
6.4 Proposed System	33
7 RESULTS AND DISCUSSION	37
8 TESTING	39
9 CONCLUSION AND FUTURE SCOPE	41
9.1 Conclusion	41
9.2 Future Scope	41
REFERENCES	43

LIST OF FIGURES

	Page No.
Fig 1.1 Agile Methodology	5
Fig 1.2 Scrum	6
Fig 3.1 Data Visualization	15
Fig 3.2 Data Splitting	19
Fig 4.1 Training Model Process	24
Fig 4.2 Use Case	25
Fig 5.1 Diabetes Dataset	28
Fig 5.2 Heart Disease Dataset	29
Fig 6.1 SVM	31
Fig 6.2 KNN	32
Fig 6.3 Prediction Engine Proposed Model	33
Fig 6.4 Trained KNN, SVM and Naïve Bayes Models	34
Fig 6.5 Pre-processed Diabetes Dataset	34
Fig 6.6 Pre-processed Heart Disease Dataset	35
Fig 6.7 Returning trained models to the API's on function calls	35
Fig 6.8 API to predict Diabetes	36
Fig 6.9 API to predict Heart Disease	36
Fig 7.1 Prediction Engine Result Page	37
Fig 7.2 Diabetes Form to collect User Data for Prediction	38
Fig 7.3 Heart Disease Form to collect User Data for Prediction	38
Fig 8.1 Form Validation and API Testing	39
Fig 8.2 Trained Models tested for Diabetes Prediction	40
Fig 8.3 Trained Models tested for Heart Disease Prediction	40

CHAPTER 1

INTRODUCTION

Use of computing in the field of medicine can be seen from the early 1950s. However, the first applications of AI in medicine can only be seen during the 1970s through expert systems such as INTERNIST-I, MYCIN, ONCOSIN.

The application of artificial intelligence in medicine was mostly limited in the US before 1980. An international conference was organized on 13-14 September 1985 in Italy. With the aim of creating an active research community, in 1986 “Society for Artificial Intelligence in Medicine” was established which biennially organizes international conferences.

One of the problems associated with using artificial intelligence in medicine is unavailability of data, which can be solved using Electronic Medical Records (EMR). The concept of EMR was introduced by Larry Weed during the late 1960s. The US government started using EMR in the 1970s with the Department of Veteran Affairs.

“Knowledge Engineering” was the major research theme during the 1980s, but after 1990 the research shifted to various topics. After 2000, “Machine Learning and Data Mining” has been the major research theme.

Types of Diabetes

1) Type one diabetes outcomes due to the failure of the pancreas to supply enough hypoglycaemic agent. This type was spoken as "insulin-dependent polygenic disease mellitus" (IDDM) or "juvenile diabetes". The reason is unidentified. The type one polygenic disease found in children beneath twenty years old. People suffer throughout their life because of the type one diabetic and rest on insulin vaccinations. The diabetic patients must often follow workouts and fit regime which are recommended by doctors.

2) The type two diabetes starts with hypoglycaemic agent resistance, a situation in which cells fail to respond the hypoglycaemic agents efficiently. The sickness develops due to the absence of hypoglycaemic agent that additionally built. This type was spoken as "non-insulin-dependent polygenic disease mellitus". The usual cause is extreme weight. The quantity of people affected by type two will be enlarged by 2025. The existence of diabetes mellitus is condensed by 3% in rural zone as compared to the urban zone. The prehypertension is joined with bulkiness, obesity and diabetes mellitus. The study found that an individual United Nations agency has traditional vital sign.

3) Type 3 Gestational diabetes occurs when a woman is pregnant and develops high blood sugar levels without a previous history of diabetes. Therefore, it is found that in total 18% of women in pregnancy have diabetes. So in the older age there is a risk of emerging gestational diabetes in pregnancy. Obesity is one of the main reasons for type-2 diabetes. The type-2 polygenic disease are under control by proper workout and taking appropriate regime. When the aldohexose level isn't reduced by the higher strategies then medications are often recommended. The polygenic disease static report says that 29.1 million people of the United States inhabitants has diabetes.

1.1 Relevance of the Project

The major problem with using AI for the diagnosis of disease is the lack of data for training predictive models. Though there is vast amount of data including mammograms, genetic tests, and medical records, they are not open to the people who can make use of them for research. Some initiatives like “100,000 Genomes Project” in the UK, the U.S. Department of Veteran Affairs’ “Million Veteran Program”, and the NIH’s “The Cancer Genome Atlas” will hopefully provide data to researchers and data scientists.

In many countries' health records are being digitized. The adoption of EMR is also increasing. According to a data brief by The Office of National Coordinator for Health Information Technology (ONC), 3 out of 4 private or not-for-profit hospitals adopted at least a Basic EHR system in the US. In many other countries, different EHR systems exist. The Stockholm EPR corpus is a great example of such systems which consists of data from 512 clinical units with over 2 million patient records. India is thinking about setting up a National eHealth Authority (NeHA) during the Digital India program. These types of electronic health documents provide a huge amount of data for intelligence data analysis. Many research have been conducted on predicting various diseases like Liver Disease, Heart Disease, Diabetes etc., detecting tumours, leukaemia etc. using computer vision, assisting doctors in making efficient decisions.

1.2 Problem Statement

The primary goal is to develop a prediction engine which will allow the users to check whether they have diabetes or heart disease sitting at home. The user need not visit the doctor unless he has diabetes or heart disease, for further treatment. The prediction engine requires a large dataset and efficient machine learning algorithms to predict the presence of the disease. Pre-processing the dataset to train the machine learning models, removing redundant, null, or invalid data for optimal performance of the prediction engine.

Doctors rely on common knowledge for treatment. When common knowledge is lacking, studies are summarized after some number of cases have been studied. But this process takes time, whereas if machine learning is used, the patterns can be identified earlier. For using machine learning, a huge amount of data is required. There is very limited amount of data available depending on the disease. Also, the number of samples having no diseases is very high compared to the number of samples having the disease. This project is about performing two case studies to compare the performance of

various machine learning algorithms to help identify such patterns in (i) and to create a platform for easier data sharing and collaboration.

1.3 Objectives

The primary aim of this project is to analyze the “Pima Indian Diabetes Dataset” and “Cleveland Heart Disease Dataset” and use Support Vector Machine, Naïve Bayes, and K-Nearest Neighbors for prediction.

The secondary aim is to develop a web application that allows users to predict heart disease and diabetes utilizing the prediction engine.

1.4 Scope of the Project

The disease diagnosis system will permit end-users to predict heart disease and diabetes.

1.4.1 Growth of AI Systems

Artificial Intelligence is one of the hottest topics today. The revenue for cognitive and artificial intelligence systems is expected to hit \$12.5 billion.

1.4.2 Availability of Doctors and Chatbots

Other than disease diagnosis, artificial intelligence can be used to streamline and optimize the clinical process. There is only one doctor for over 1600 patients in India .AI health assistants can help in covering large part of clinical and outpatient services freeing up doctor’s time to attend more critical cases. Chatbots like “Your.MD” can assist patients by understanding patients’ symptoms and suggest easy-to-understand medical information about their condition. Other assistants like “Ada” integrated with

“Amazon Alexa” provides a detailed symptom assessment report and also provides an option to contact a real doctor. Such assistants make use of Natural Language Processing and Deep Learning to understand the user and generate suggestions.

1.4.3 Internet of Things (IoT), Healthcare and Machine Learning

Increasing use of Internet of Things has promising benefits in healthcare.

Dynamically collecting patient data using remote sensors can help in early detection of health problems and aid in preventive care.

1.5 Project Methodology

Agile:

Agile is a process by which a team can manage a project by breaking it up into several stages and involving constant collaboration with stakeholders and continuous improvement and iteration at every stage. It promotes **continuous iteration** of development and testing throughout the software development life cycle of the project. Both development and testing activities are concurrent.

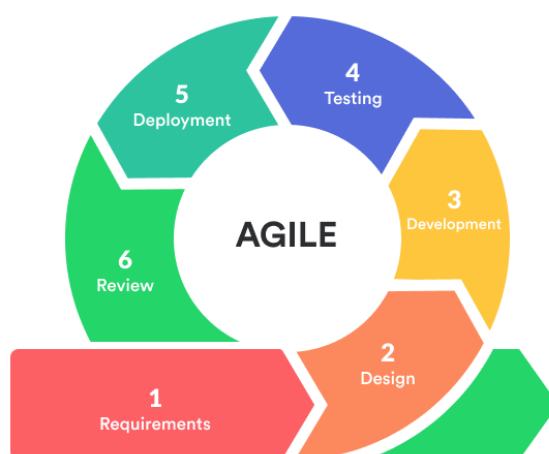


Fig 1.1 - Agile Methodology

Scrum

SCRUM is an agile development method which concentrates specifically on how to manage tasks within a team-based development environment. Scrum encourages teams to learn through experiences, self-organize while working on a problem.

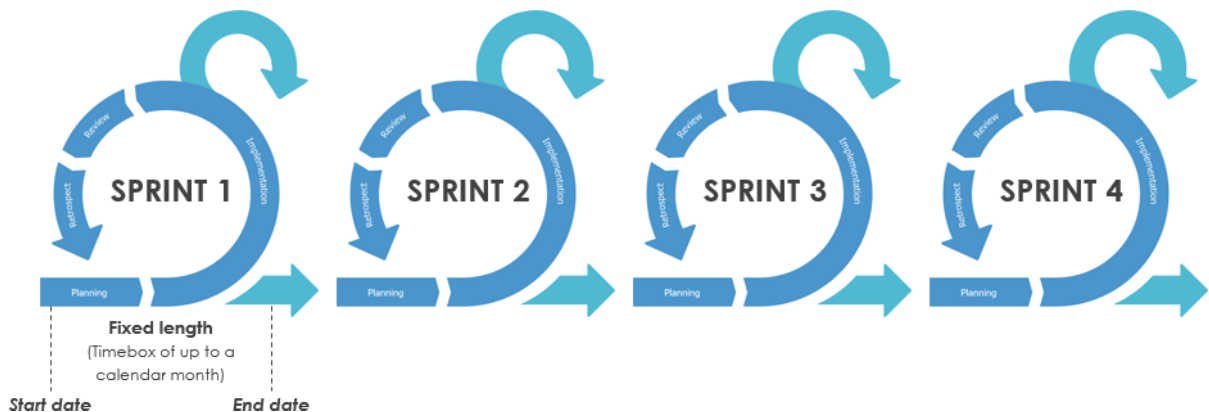


Fig 1.2 – Scrum

The Main Artefacts

- **Product Backlog** is the master list of work that needs to get done maintained by the product owner or product manager.
- **Sprint Backlog** is the list of items, user stories, or bug fixes, selected by the development team for implementation in the current sprint cycle.
- **Increment** (or Sprint Goal) is the usable end-product from a sprint.

Sprint 1	Sprint 2	Sprint 3	Sprint 4	Sprint 5	Sprint 6
August 2019	September 2019- October 2019	November 2019	December 2019	January 2020- February 2020	March 2020- April 2020
Discussion of project selection	Literature survey, Synopsis, SDD, Research Notes	Freezing of algorithm, framework and software	Exams (hiatus)	Coding and implementation	Testing and adding last minute features

Table 1.1 – Sprint

CHAPTER 2

LITERATURE SURVEY

The following chapters give an overview of the various methodologies used by various authors for disease prediction using machine learning methodologies. We can observe that there is fine comparison made between 5 major machine learning algorithms whether they are able to predict the presence of the disease with a greater accuracy, achieving optimal performance. The research efforts presented by the authors in the following papers are focused in developing and evaluating a web-based tool for disease prediction

2.1 Diabetes Prediction Using Different Machine Learning Approaches

Author: Priyanka Sonar, Prof. K. JayaMalini

Published In: Proceedings of the Third International Conference on Computing Methodologies and Communication (ICCMC 2019)

The authors have used Machine Learning approaches to predict diabetes [1]

Diabetes is one of lethal diseases in the world. It is additional an inventor of various varieties of disorders for example: coronary failure, blindness, urinary organ diseases etc. In such a case the patient is required to visit a diagnostic centre, to get their reports after consultation. Due to every time they must invest their time and currency. But with the growth of Machine Learning methods we have got the flexibility to search out an answer to the current issue, we have got advanced system mistreatment information processing that has the ability to forecast whether the patient has polygenic illness or not. Furthermore, forecasting the sickness initially ends up in providing the patients before it begins vital. Information withdrawal has the flexibility to remove unseen data

from a large quantity of diabetes associated information. The aim of this analysis is to develop a system which might predict the diabetic risk level of a patient with a better accuracy. Model development is based on categorization methods as Decision Tree, ANN, I Bayes and SVM algorithms. For Decision Tree, the models give precisions of 85%, for I Bayes 77% and 77.3% for Support Vector Machine. Outcomes show a significant accuracy of the methods.

Conclusion: SVM is very good when we have no idea on the data. Even with unstructured and semi structured data like text, images, and trees SVM algorithm works well. The drawback of the SVM algorithm is that to achieve the best classification results for any given problem, several key parameters are needed to be set correctly. **Decision tree:** It is easy to understand and rule decision tree. Instability is there in decision tree, that is bulky change can be seen by minor modification in the data structure of the optimal decision tree. They are often relatively inaccurate. **I Bayes:** It is robust, handles the missing values by ignoring probability estimation calculation. Sensitive to how inputs are prepared. Prone bias when increase the number of training dataset. **ANN:** Gives good prediction and easy to implement. Difficult with dealing with big data with complex models. Require huge processing time.

2.2 Implementation of a Web Application to Predict Diabetes Disease: An Approach Using Machine Learning Algorithm,

Authors: Samrat Kumar Dey, Ashraf Hossain and Md. Mahbubur Rahman

Published In: 2018 21st International Conference of Computer and Information Technology (ICCIIT)

The authors design and develop a web application to predict diabetes [2]

Diabetes is caused due to the excessive amount of sugar condensed into the blood. Currently, it is considered as one of the lethal diseases in the world. People all around the globe are affected by this severe disease knowingly or unknowingly. Other diseases

like heart attack, paralyzed, kidney disease, blindness etc. are also caused by diabetes. Numerous computer-based detection systems were designed and outlined for anticipating and analysing diabetes. Usual identifying process for diabetic patients needs more time and money. But with the rise of machine learning, we have that ability to develop a solution to this intense issue. Therefore, we have developed an architecture which has the capability to predict where the patient has diabetes or not. Our main aim of this exploration is to build a web application based on the higher prediction accuracy of some powerful machine learning algorithm. We have used a benchmark dataset namely Pima Indian which can predict the onset of diabetes based on diagnostics manner. With an accuracy of 82.35% prediction rate Artificial Neural Network (ANN) shows a significant improvement of accuracy which drives us to develop an Interactive Web Application for Diabetes Prediction.

Conclusion: In this paper, authors proposed a web-based application for the successful prediction of Diabetes Diseases. From different machine learning algorithms Artificial Neural Network (ANN) provides highest accuracy with Min Max Scaling Method on Indian Pima Dataset. As they have proposed and developed an approach for diabetes disease prediction using machine learning algorithm, it has significant potential in the field of medical science for the detection of various medical data accurately. In the near future, focus is to use a deep learning model and prepare a Location based Dataset from real medical data for the successful prediction of diabetes disease

2.3 Diabetes Disease Prediction Using Data Mining

Authors: Deeraj Shetty, Kishor Rit, Sohail Shaikh and Nikita Patil

Published In: 2017 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS)

The author talks about the significance of dataset in prediction of the disease.

Data mining is a subfield in the subject of software engineering. It is the methodical procedure of finding examples in huge data sets including techniques at the crossing

point of manufactured intelligence, machine learning, insights, and database systems. The goal of the data mining methodology is to think data from a data set and change it into a reasonable structure for further use. Our examination concentrates on this part of Medical conclusion learning design through the gathered data of diabetes and to create smart therapeutic choice emotionally supportive network to help the physicians. The primary target of this examination is to assemble Intelligent Diabetes Disease Prediction System that gives analysis of diabetes malady utilizing diabetes patient's database. In this system, we propose the use of algorithms like Bayesian and KNN (K-Nearest Neighbour) to apply on diabetes patient's database and analyse them by taking various attributes of diabetes for prediction of diabetes disease.

Expected Result: The goal of our project is to know whether the patient is diabetic or not, patient will be diagnosed and it will be depending on the attributes that we are going to take, such as age, pregnancy, pg concentration, tri fold thick, serum ins, body mass index (bmi), dp function, diastolic bp i.e. the factors which are majorly responsible for diabetes. So, to reduce the correctly know whether the patient is diabetic or not, we are developing a system which will be a prediction system for diabetes patients. Another best thing about the system is it is will give accurate results whether the patient is diabetic or not with the help of the knowledge base of the larger dataset that we are going to use added the recommendations we are going to provide based on the diabetic levels of the patients. Also, the prediction of the disease will be done with the help of Bayesian algorithm and K-NN algorithm.

Conclusion: By our in-depth analysis of literature survey, we acknowledged that the prediction done earlier did not use a large dataset. A large dataset ensures better prediction. Also, what it lacks is recommendation system. When they predict they give some recommendations to the patient on how to control or prevent diabetes in case of minor signs of diabetes. The recommendations would be such, that when followed it will help the patient. Thus build up a system which will anticipate diabetic patient with the assistance of the Knowledge base which we have of dataset of around 2000 diabetes

patients and furthermore to give suggestions on the premise of the nearness of levels of diabetes patients. Prediction will be done with the help of two algorithms Naïve Bayes and K-Nearest Neighbour and also compare which algorithm gives better accuracy on the basis of their performance factors. This system which will be developed can be used in HealthCare Industry for Medical Check of diabetes patients.

2.4 An Intelligent System for Diabetes Prediction

Authors: Zhilbert Tafa, Nerxhivane Pervetica and Bertran Karahoda

Published In: 4thMediterranean Conference on Embedded Computing MECO – 2015 Budva, Montenegro

The authors develop an intelligent system for diabetes prediction.

With the emerging increase of diabetes, that recently affects around 346 million people, of which more than one-third go undetected in early stage, a strong need for supporting the medical decision-making process is generated. A number of research have focused either in using one of the algorithms or in the comparisons of the performances of algorithms on a given, usually predefined and static datasets that are accessible through the Internet. This paper focuses on the joint implementation of the support vector machine (SVM) and Naïve Bayes statistical analysing, in the dataset acquired from the medical examinations of 402 patients, to improve the computer-supported diagnosis reliability. The dataset contains some attributes that have not been previously used in computer-based evaluations. The results show that the joint implementation of two algorithms significantly improves the overall reliability of the system outcome, which is crucial in the computer-supported diabetes diagnostic process.

Conclusions and Future Work: The research efforts presented in this paper are focused in developing and the evaluation of a computer-based support tool for the diabetes detection. The presented approach is based on the joint implementation of two algorithms in Matlab that have been executed on the newly acquired dataset with the different attributes as compared to the previous work in this field. The algorithms are

executed and evaluated independently but the decision making is based on the joint outcomes from both algorithms. The aim of this approach is to make the decision more reliable. As shown in the paper, both SVM and naïve Bayes algorithm have individually shown high overall classifier performances of 95, 52% and 94, 52%, respectively. The joint implementation on the same, newly added record leads to one of the three answers: a) the patient is diagnosed with diabetes (or pre-diabetic condition), b) the patient is not diagnosed as having the mentioned condition, and c) the patient is further directed to the additional clinical examinations. If two algorithms show different results, the answer is classified as condition c). Otherwise, the accuracy of the answers a) or b), as shown in the paper, is improved up to the value of 97,6%. The presented methodology minimizes the false negative answers, which is a crucial issue in medical diagnoses. Finally, the construction approach, the architecture, and the evaluation of a diabetes classification tool presented in this paper, should provide an important guideline to further construction of the similar applications on improving and helping the decision making process in disease detection. The development of a user-friendly and widely accessible application would enable the personal self-screening on diabetic or pre-diabetic condition which is crucial to the disease treatment performance. The future work will focus on further quantitative evaluations of the developed tool regarding the extensive clinical examinations and results. Also, other methods should be involved in finding the best fit in the sense of accuracy, processing time, etc. The influence of cultural-related biases (such as those related to the nutrition structure and habits) should also be minimized for the results to get more generalized.

2.5 Algorithms and Results across Publications

In 2016, Dey et al. used Principal Component Analysis (PCA) for selection of features and analysed the performance of heart disease prediction using Naïve Bayes, Decision Trees and Support Vector Machines. After reducing the correlated variables to linearly uncorrelated variables using PCA, SVM outperformed Naïve Bayes and Decision

Trees. The authors recommend building a desktop web application using SVM to predict heart disease.

After analysing multiple studies in the area of heart disease prediction Darne et al. (2016) suggests the following:

- Neural Networks and Fuzzy techniques help in better prediction.
- Ant Colony Optimization, Support Vector Machine can be used for better prediction.
- K-means clustering algorithm can be used to resolve data overfitting and overgeneralization.

In 2016, Purushottam et al. proposed an efficient heart disease prediction system using decision trees. They presented the rules generated through the experiment which were further pruned and duplicates were removed and class wise rule sets were prepared. The results were tested using 10-fold cross validation. This system outperformed SVM, MLP and various other algorithms with an accuracy of 86.7%.

In 2016, Srikanth & Deverapalli performed a critical study of classification algorithms namely Decision trees and Naïve Bayes. They used the Pima Indian Diabetes dataset and found the results to be accurate. They suggest enhancing the dataset using Linear regression and Logistic regression models.

In 2016, El-Baz et al. proposed a classification system using combined MLP and combined CFBN. They used the Pima Indian Diabetes Dataset. At first, they used MLP and CFBN individually and obtained an accuracy of 80.21% and 81.77% respectively. Later, when they used combined techniques, the accuracy was higher than both the individual classifiers – 95.31% and 96.88% respectively.

CHAPTER 3

SYSTEM REQUIREMENTS SPECIFICATION

3.1 Functional Requirements – Dataset preparation and pre processing

3.1.1 Data Collection

Data collection is defined as the procedure of collecting, measuring and analysing accurate insights for research using standard validated techniques. A researcher can evaluate their hypothesis based on collected data. In most cases, data collection is the primary and most important step for research, irrespective of the field of research. The approach of data collection is different for different fields of study, depending on the required information. The most critical objective of data collection is ensuring that information-rich and reliable data is collected for statistical analysis so that data-driven decisions can be made for research.

3.1.2 Data Visualization

Data visualization is the graphical representation of information and data. By using visual elements like charts, graphs, and maps, data visualization tools provide an accessible way to see and understand trends, outliers, and patterns in data. Example -

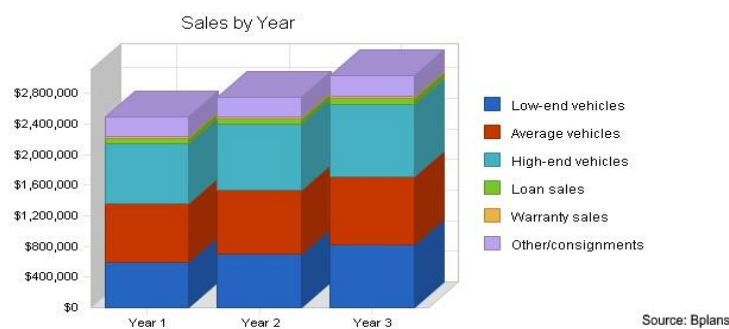


Fig 3.1 – Data Visualization

3.1.3 Data Labelling

Supervised machine learning, which we'll talk about below, entails training a predictive model on historical data with predefined target answers. An algorithm must be shown which target answers or attributes to look for. Mapping these target attributes in a dataset is called labelling. Data labelling takes much time and effort as datasets sufficient for machine learning may require thousands of records to be labelled. For instance, if your image recognition algorithm must classify types of bicycles, these types should be clearly defined and labelled in a dataset.

3.1.4 Data Selection

Data selection is defined as the process of determining the appropriate data type and source, as well as suitable instruments to collect data. Data selection precedes the actual practice of data collection. This definition distinguishes data selection from selective data reporting (selectively excluding data that is not supportive of a research hypothesis) and interactive/active data selection (using collected data for monitoring activities/events, or conducting secondary data analyses). The process of selecting suitable data for a research project can impact data integrity. After having collected all information, a *data analyst* chooses a subgroup of data to solve the defined problem. For instance, if you save your customers' geographical location, you don't need to add their cell phones and bank card numbers to a dataset. But purchase history would be necessary. The selected data includes attributes that need to be considered when building a predictive model.

3.1.5 Data Pre-processing

Data pre-processing is a data mining technique that involves transforming raw data into an understandable format. Real-world data is often incomplete, inconsistent, and/or lacking in certain behaviours or trends, and is likely to contain many errors. Data pre-processing is a proven method of resolving such issues. The purpose of pre-processing

is to convert raw data into a form that fits machine learning. Structured and clean data allows a data scientist to get more precise results from an applied machine learning model. The technique includes data formatting, cleaning, and sampling.

Data formatting. The importance of data formatting grows when data is acquired from various sources by different people. The first task for a data scientist is to standardize record formats. A specialist checks whether variables representing each attribute are recorded in the same way. Titles of products and services, prices, date formats, and addresses are examples of variables. The principle of data consistency also applies to attributes represented by numeric ranges.

Data cleaning. This set of procedures allows for removing noise and fixing inconsistencies in data. A data scientist can fill in missing data using imputation techniques, e.g. substituting missing values with mean attributes. A specialist also detects outliers — observations that deviate significantly from the rest of distribution. If an outlier indicates erroneous data, a data scientist deletes or corrects them if possible. This stage also includes removing incomplete and useless data objects.

Data anonymization. Sometimes a data scientist must anonymize or exclude attributes representing sensitive information (i.e. when working with healthcare and banking data).

Data sampling. Big datasets require more time and computational power for analysis. If a dataset is too large, applying data sampling is the way to go. A data scientist uses this technique to select a smaller but representative data sample to build and run models much faster, and at the same time to produce accurate outcomes.

3.1.6 Data Transformation

Data transformation is the process of converting data from one format or structure into another format or structure. Data transformation is critical to activities such as data

integration and data management. Data transformation can include a range of activities: you might convert data types, cleanse data by removing nulls or duplicate data, enrich the data, or perform aggregations, depending on the needs of your project.

Scaling. Data may have numeric attributes (features) that span different ranges, for example, millimetres, meters, and kilometres. Scaling is about converting these attributes so that they will have the same scale, such as between 0 and 1, or 1 and 10 for the smallest and biggest value for an attribute.

Decomposition. Sometimes finding patterns in data with features representing complex concepts is more difficult. Decomposition technique can be applied in this case. During decomposition, a specialist converts higher level features into lower level ones. In other words, new features based on the existing ones are being added. Decomposition is mostly used in time series analysis. For example, to estimate a demand for air conditioners per month, a market research analyst converts data representing demand per quarters.

Aggregation. Unlike decomposition, aggregation aims at combining several features into a feature that represents them all. For example, you have collected basic information about your customers and particularly their age. To develop a demographic segmentation strategy, you need to distribute them into age categories, such as 16-20, 21-30, 31-40, etc. You use aggregation to create large-scale features based on small-scale ones. This technique allows you to reduce the size of a dataset without the loss of information.

3.1.7 Data Splitting

A dataset used for machine learning should be partitioned into three subsets — training, test, and validation sets.

Training set. A *data scientist* uses a training set to train a model and define its optimal parameters — parameters it must learn from data.

Test set. A test set is needed for an evaluation of the trained model and its capability for generalization. The latter means a model’s ability to identify patterns in new unseen data after having been trained over a training data. It is crucial to use different subsets for training and testing to avoid model overfitting, which is the incapacity for generalization we mentioned above.

Validation set. The purpose of a validation set is to tweak a model’s hyperparameters — higher-level structural settings that cannot be directly learned from data. These settings can express, for instance, how complex a model is and how fast it finds patterns in data.

The proportion of a training and a test set is usually 80 to 20 percent, respectively. A training set is then split again, and its 20 percent will be used to form a validation set. At the same time, machine learning practitioner Jason Brownlee suggests using 66 percent of data for training and 33 percent for testing. A size of each subset depends on the total dataset size

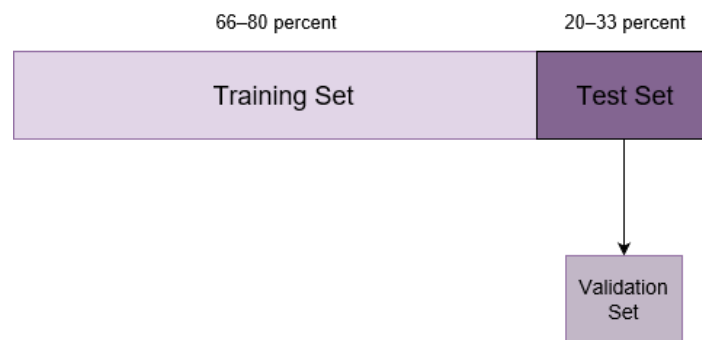


Fig 3.2 – Data Splitting

The more training data a data scientist uses, the better the potential model will perform. Consequently, more results of model testing data lead to better model performance and generalization capability.

3.1.8 Modelling

After pre-processing the collected data and split it into three subsets, we can proceed with a model training. This process entails “feeding” the algorithm with training data. An algorithm will process data and output a model that is able to find a target value (attribute) in new data — an answer you want to get with predictive analysis. The purpose of model training is to develop a model.

Two model training styles are most common — supervised and unsupervised learning. The choice of each style depends on whether you must forecast specific attributes or group data objects by similarities.

Model evaluation and testing: The goal of this step is to develop the simplest model able to formulate a target value fast and well enough. A data scientist can achieve this goal through model tuning. That is the optimization of model parameters to achieve an algorithm’s best performance. One of the more efficient methods for model evaluation and tuning is cross-validation.

Cross-validation: Cross-validation is the most used tuning method. It entails splitting a training dataset into ten equal parts (folds). A given model is trained on only nine folds and then tested on the tenth one (the one previously left out). Training continues until every fold is left aside and used for testing. As a result of model performance measure, a specialist calculates a cross-validated score for each set of hyperparameters. A data scientist trains models with different sets of hyperparameters to define which model has the highest prediction accuracy. The cross-validated score indicates average model performance across ten hold-out folds.

3.1.9 Model Deployment

Deployment is the method by which you integrate a machine learning model into an existing production environment to make practical business decisions based on data. It is one of the last stages in the machine learning life cycle and can be one of the most cumbersome. Often, an organization's IT systems are incompatible with traditional model-building languages, forcing data scientists and programmers to spend valuable time and brainpower rewriting them.

3.2 Non-Functional Requirements

3.2.1 Usability

The system should be easy to use. The system also should be user friendly for users because anyone can use it instead of programmers.

3.2.2 Reliability

This software will be developed with machine learning, feature engineering and deep learning techniques. So, in this step there is no certain reliable percentage that is measurable. Also, user provided data will be used to compare with results and measure reliability. With recent machine learning techniques, user gained data should be enough for reliability if enough data is obtained.

3.2.3 Performance

Processing time and response time should be as little as possible providing the result at a faster rate when compared to other methods.

3.2.4 Supportability

The system should require Python knowledge to maintenance. If any problem acquire in user side and deep learning methods, it requires code knowledge and deep learning background to solve.

3.3 Hardware Requirements

- OS: Windows 10
- RAM: Minimum of 4GB

3.4 Software Requirements

- Basic Text-Editor: Visual Studio Code, Pycharm.
- Jupyter Notebook: Development of Python Scripts.
- Flask Framework: API Development.
- Angular Framework: UI Development.

3.4.1 Why Python?

- General purpose programming language
- Increasing popularity for use in data science
- Easy to build end-to-end products like web applications
- Since the goal of this project is to build a web application, Python is a better choice. Though frameworks like Shiny can be used with R to create web applications, it is extremely slow.

There are two major libraries for machine learning in python: TensorFlow, ScikitLearn. The main differences are discussed below.

TensorFlow	Scikit-Learn
Low-Level, algorithms should be implemented manually	Provides off-the-shelf algorithms
Better choice for deep learning	Less efficient for deep learning
Steep learning curve	Easier to learn

Table 3.1 – Machine Learning Libraries

Scikit-learn was chosen for the project for the following reasons:

- Off-the-shelf algorithms
- Shallow learning curve compared to TensorFlow

CHAPTER 4

SYSTEM ANALYSIS AND DESIGN

4.1 Initial Engine Design

Before writing any code, the initial design of the algorithm for the prediction engine was created using Microsoft Visio. The initial design can be seen in Figure below.

The below algorithm is repeated for all classification algorithms. Furthermore, the accuracy score is calculated using 10-fold cross validation.

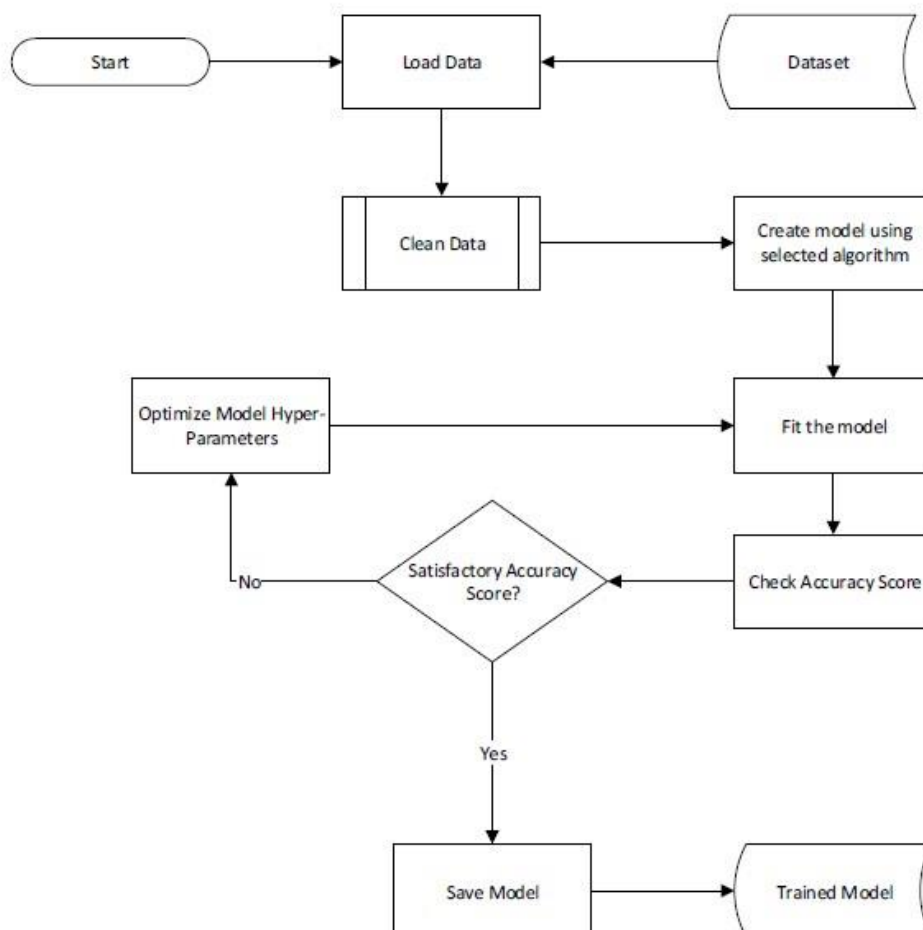


Fig 4.1 – Training Model Process

4.2 Use Cases

Though use cases are not recommended by agile development or feature driven development, use cases were created to better understand the requirements of the system. The use cases are packaged according to domain areas. Detailed description of only some of the use cases is present in this section

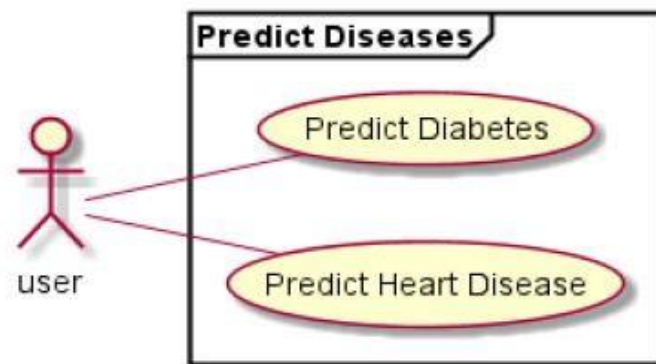


Fig 4.2 – Use Case

The description of use cases for predicting diseases shown in Figure 4.2 is given below.

Use Case Number: 1

Title: Predict Heart Disease

Description: The user submits the form with the values obtained after performing various medical tests. Upon submission, the user receives message indicating whether they have a risk of developing heart disease.

Actors: User

Priority: Essential

Postconditions: Patient successfully receives the prediction results.

Main Success Scenario:

1. The user performs all the tests and enters valid test results
2. The system performs prediction using the prediction engine API.
3. The user receives the prediction results.

Use Case Number: 2

Title: Predict Diabetes

Description: The user submits the form with the values obtained after performing various medical tests. Upon submission, the user receives a message indicating whether they have a risk of developing diabetes.

Actors: User

Priority: Essential

Postconditions: Patient successfully receives the prediction results.

Main Success Scenario:

1. The user performs all the tests and enters valid test results.
2. The system performs prediction using the prediction engine API.
3. The user receives the prediction results

Alternative Scenario:
If the user enters invalid value for any of the provided form fields: the system will display an error message.

If the system is unable to communicate with the prediction engine API: the system will display an error message.

CHAPTER 5

PROBLEM FORMULATION

The primary goal is to develop a prediction engine which will allow the users to check whether they have diabetes or heart disease sitting at home. The user need not visit the doctor unless he has diabetes or heart disease, for further treatment. The prediction engine requires a large dataset and efficient machine learning algorithms to predict the presence of the disease. Pre-processing the dataset to train the machine learning models, removing redundant, null or invalid data for optimal performance of the prediction engine.

The secondary aim is to develop a web application that allows users to predict heart disease and diabetes utilizing the prediction engine

Why classification supervised learning?

Since the project involves identifying whether there is risk of diabetes or heart diseases, it is obviously a classification problem. The “Pima Indian Diabetes Dataset” and “Cleveland Heart Disease Dataset” datasets used for this project already have the outcome hence supervised learning.

5.1 Datasets Formulation

In this section, the “Pima Indian Diabetes” and the “Cleveland Heart Disease” datasets have been discussed.

5.1.1 The “Pima Indian Diabetes” Dataset

The Pima Indian Diabetes Dataset has been taken from the UCI machine-learning repository (Lichman, 2013). The dataset contains 768 records of female patients among which 500 are non-diabetic and 268 are diabetic.

S.N.	Attribute	Description	Value
1	Pregnant	Number of times pregnant	Continuous
2	PlasmaGlucoseConc	Plasma glucose concentration of 2 hours in an oral glucose tolerance test	Continuous
3	DiastolicBP	Diastolic Blood Pressure in mmHg	Continuous
4	TSFT	Triceps Skin Fold Thickness measured in mm.	Continuous
5	2hrSerumInsuloin	2-Hour serum insulin measured in $\mu\text{U/ml}$	Continuous
6	BMI	Body Mass Calculated using: $\frac{\text{Weight in kg}}{(\text{height in meter})^2}$	Continuous
7	DPF	Diabetic Pedigree function – how likely the person is to have given their family history and other factors. See Appendix F below	Continuous
8	Age	Age in years	Continuous
9	Res	Presence of diabetes	1 – Yes 0 – No

Fig 5.1 – Diabetes Dataset

5.1.2 The “Cleveland Heart Disease” Dataset

The Cleveland Heart Disease Dataset has been taken from the UCI machine-learning repository (Lichman, 2013). The dataset contains 303 records. The dataset contains 76 attributes, but all the studies have only considered a subset of 14 attributes:

S.N.	Attribute	Description	Value
1	Age	Age in years	Continuous
2	Sex	Male or Female	0 – Female 1 – Male
3	Cp	Chest Pain Type	1 – Typical Angina 2 – Atypical Angina 3 – Non-angina 4- Asymptomatic
4	Threstbps	Resting Blood Pressure in mmHg	Continuous
5	Chol	Serum cholesterol in mg/dl	Continuous
6	Fbs	Fasting blood sugar	1 – fbs > 120 mg/dl 0 – fbs < 120 mg/dl
7	Restecg	Resting electrocardiographic results	0 – Normal 1 – Having ST-T wave abnormality 2 – Showing probable or definite left ventricular hypertrophy by Estes' criteria
8	Thalach	Maximum heart rate achieved	Continuous
9	Exang	Exercise induced angina	1 – Yes 0 – No
10	Oldpeak	ST depression induced by exercise relative to rest	Continuous
11	Slope	The slope of the peak exercise ST segment	1 – Upsloping 2 – Flat 3 – Down sloping
12	Ca	Number of major vessels colored by fluoroscopy	0 to 3
13	Thal	Result of Thallium Heart Scan	3 – Normal 6 – Fixed Defect 7 – Reversible Defect
14	Num	Presence of heart disease	0 – Absent 1 to 3 – Present

Fig 5.2 – Heart Disease Dataset

CHAPTER 6

IMPLEMENTATION

6.1 Naïve Bayes

Naive Bayes is a set of supervised learning algorithms based on the Bayes' theorem with the “naïve” assumption of independence between every pair of features. Despite its simplicity, it often outperforms more sophisticated classification methods.

If there are input variables x and output variable y , Bayes' theorem states the following relationship.

$$p(y|x) = p(y) \cdot p(x|y) / p(x)$$

In this project, Gaussian Naïve Bayes algorithm has been implemented. In case of Gaussian Naïve Bayes, the likelihood of the features is assumed to be Gaussian i.e. all continuous values x associated with class y are distributed according to Gaussian distribution.

Given a continuous attribute x in training data, the data is first segmented by the class y . Then, the mean and variance of x in each class is computed. If μ be the mean of the values in x associated with class y , then let σ^2 be the variance of the values in x associated with class y . Suppose there is some observation value v then, the probability distribution of v given by class y , $p(x=v | y)$, can be computed by plugging into the equation for a normal distribution

6.2 Support Vector Machine

Support vector machines (SVMs) are a set of supervised learning methods used for classification, regression and outliers' detection. It is effective when the number of

attributes is greater than the number of samples but when that number increases significantly, this method is likely to give poor performance.

For given labelled training data, SVM outputs an optimal hyper plane to categorize new examples. The optimal hyper plane maximizes the margin of the training data as shown below in Figure

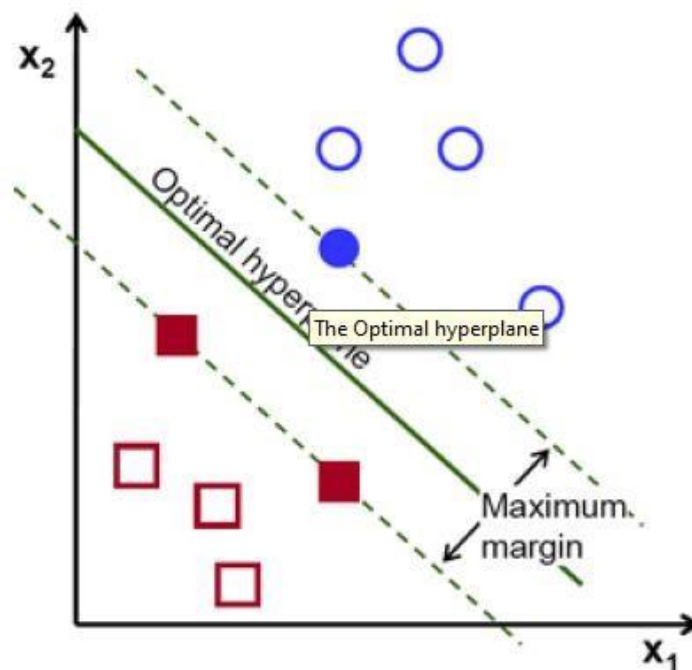


Fig 6.1 - SVM

6.3 K Nearest Neighbours

K-Nearest Neighbours Classification is a type of instance-based learning or non-generalizing learning which stores instances of the training data. It predicts a new point by finding a predefined number of training samples closest in distance.

Weights can also be given for classifying a new point. In scikit-learn, each neighbour can either be given uniform weight or can be given weights proportional to the inverse of the distance from the query point or a user defined weight.

An example of K-Nearest Neighbours classifier is given below in Figure using uniform weight and distance weight in scikit-learn.

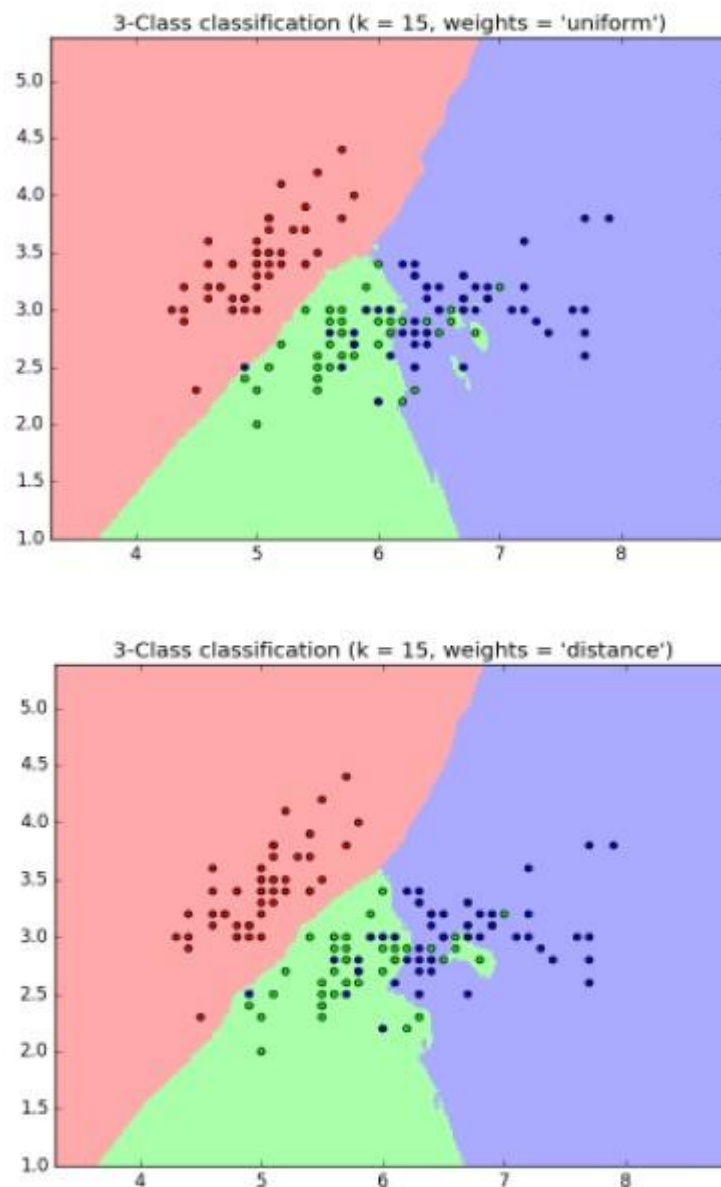


Fig 6.2 - K Nearest Neighbours

6.4 Proposed System

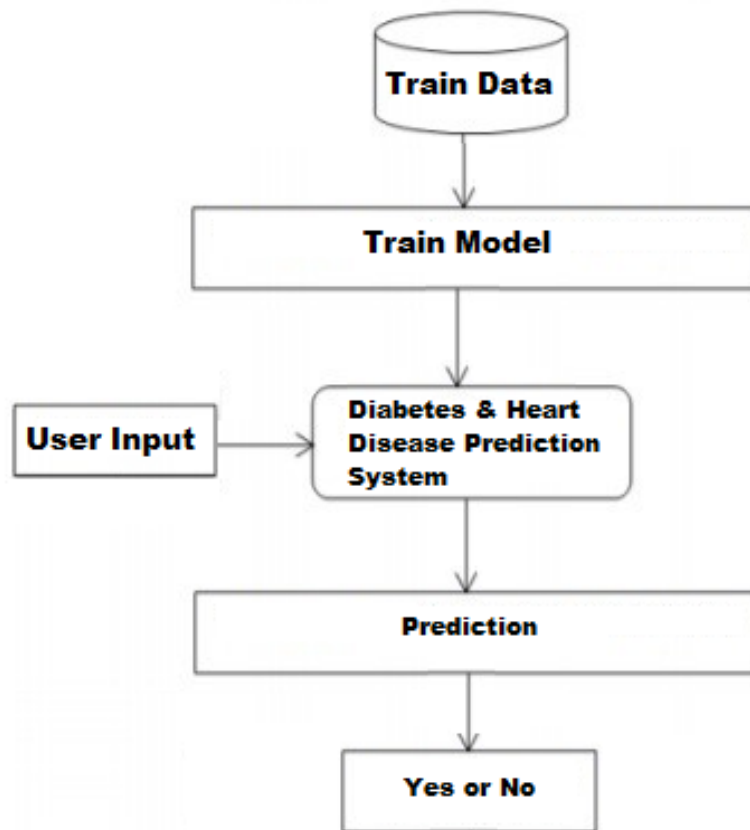


Fig 6.3 - Prediction Engine Proposed Model

- The user input is received by the web application using HTML forms.
- The Web application makes use of HTTP modules to send and receive the data to the API's.
- The API's will receive the user input in the form of a JSON object (key-value pair).
- Trained KNN, SVM and Naïve Bayes Models in the form of pickle files are consumed by the flask file housing at the local system
- The trained models are called by passing the user input JSON object
- Prediction result is sent as a response to the API calls

Trained KNN, SVM and Naïve Bayes Models were used to predict the presence of diabetes or heart disease. Pre-processed standard datasets were used to train the models post normalizing the dataset using Standard Scalar.

```
# K nearest neighbors Algorithm
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors = 24, metric = 'minkowski', p = 2)
knn.fit(X_train, Y_train)

KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
                    metric_params=None, n_jobs=None, n_neighbors=24, p=2,
                    weights='uniform')

# Support Vector Classifier Algorithm
from sklearn.svm import SVC
svc = SVC(kernel = 'linear', random_state = 42)
svc.fit(X_train, Y_train)

SVC(C=1.0, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma='scale', kernel='linear',
    max_iter=-1, probability=False, random_state=42, shrinking=True, tol=0.001,
    verbose=False)

# Naive Bayes Algorithm
from sklearn.naive_bayes import GaussianNB
nb = GaussianNB()
nb.fit(X_train, Y_train)

GaussianNB(priors=None, var_smoothing=1e-09)
```

Fig 6.4 - Trained KNN, SVM and Naïve Bayes Models

```
In [2]: # Importing dataset
dataset = pd.read_csv('diabetes.csv')
dataset
```

Out[2]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1
...
763	10	101	76	48	180	32.9	0.171	63	0
764	2	122	70	27	0	36.8	0.340	27	0
765	5	121	72	23	112	26.2	0.245	30	0
766	1	126	60	0	0	30.1	0.349	47	1
767	1	93	70	31	0	30.4	0.315	23	0

768 rows x 9 columns

Fig 6.5 - Pre-processed Diabetes Dataset

Disease Prediction Using Machine Learning

```
In [2]: df = pd.read_csv(r'heart.csv')
df

Out[2]:
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1
...
298	57	0	0	140	241	0	1	123	1	0.2	1	0	3	0
299	45	1	3	110	264	0	1	132	0	1.2	1	0	3	0
300	68	1	0	144	193	1	1	141	0	3.4	1	2	3	0
301	57	1	0	130	131	0	1	115	1	1.2	1	1	3	0
302	57	0	1	130	236	0	0	174	0	0.0	1	1	2	0

303 rows x 14 columns

Fig 6.6 - Pre-processed Hear Disease Dataset

Post training the models, these models were extracted as pickle files and are stored at a local location which is used by the flask framework to call the trained model by passing in the user input.

```
def GetJobLibFile(filepath):
    if os.path.isfile(os.path.join(dir, filepath)):
        return joblib.load(os.path.join(dir, filepath))
    return None

def GetPickleFile(filepath):
    if os.path.isfile(os.path.join(dir, filepath)):
        return pickle.load(open(os.path.join(dir, filepath), "rb"))
    return None

def GetAllClassifiersForHeart():
    return (GetLinearSVClassifierForHeart(), GetNaiveBayesClassifierForHeart(), GetKNeighborsClassifierForHeart())

def GetStandardScalarForHeart():
    return GetPickleFile(config['heart']['scalar_file'])

def GetLinearSVClassifierForHeart():
    return GetJobLibFile(config['heart']['SVC'])

def GetNaiveBayesClassifierForHeart():
    return GetJobLibFile(config['heart']['NaiveBayes'])

def GetKNeighborsClassifierForHeart():
    return GetJobLibFile(config['heart']['KNeighbors'])

## Diabetes

def GetAllClassifiersForDiabetes():
    return (GetLinearSVClassifierForDiabetes(), GetNaiveBayesClassifierForDiabetes(), GetKNeighborsClassifierForDiabetes())

def GetStandardScalarForDiabetes():
    return GetPickleFile(config['diabetes']['scalar_file'])

def GetLinearSVClassifierForDiabetes():
    return GetJobLibFile(config['diabetes']['SVC'])

def GetNaiveBayesClassifierForDiabetes():
    return GetJobLibFile(config['diabetes']['NaiveBayes'])

def GetKNeighborsClassifierForDiabetes():
    return GetJobLibFile(config['diabetes']['KNeighbors'])
```

Fig 6.7 - Returning trained models to the API's on function calls

Two API's were developed one to predict diabetes and the other to predict heart disease, using flask framework these API's were hosted locally which will be consumed by the Front-End of the Prediction Engine.

```

@app.route("/api/diabetes", methods=['GET', 'POST'])
def predict_diabetes():
    standard_scalar = GetStandardScalarForDiabetes()

    data = request.get_json(force=True)

    dataset = [{ 'Glucose': 121.48676277858587,
                 'BloodPressure': 72.40518417462486,
                 'SkinThickness': 29.153419593345687,
                 'Insulin': 155.5482233502544,
                 'BMI': 32.457463672391
               }]

    user = [{ 'Pregnancies': data['preg'],
              'Glucose': data['glu'],
              'BloodPressure': data['bp'],
              'SkinThickness': data['sk'],
              'Insulin': data['insulin'],
              'BMI': data['bmi'],
              'DiabetesPedigreeFunction': data['dpf'],
              'Age': data['age']
            }]

    dataset = pd.DataFrame(dataset)
    user = pd.DataFrame(user)

    dataset = pd.DataFrame(dataset)
    user = pd.DataFrame(user)

    user[["Glucose", "BloodPressure", "SkinThickness", "Insulin", "BMI"]] = user[
        ["Glucose", "BloodPressure", "SkinThickness", "Insulin", "BMI"]].replace(0, np.NaN)

    user["Glucose"].fillna(dataset["Glucose"], inplace=True)
    user["BloodPressure"].fillna(dataset["BloodPressure"], inplace=True)
    user["SkinThickness"].fillna(dataset["SkinThickness"], inplace=True)
    user["Insulin"].fillna(dataset["Insulin"], inplace=True)
    user["BMI"].fillna(dataset["BMI"], inplace=True)

    user_data = standard_scalar.transform(user)

    LinearSVCClassifier, NaiveBayesClassifier, KNeighborsClassifier = GetAllClassifiersForDiabetes()
    predict_result = {
        'Predictions': {
            'LinearSVC': str(LinearSVCClassifier.predict(user_data)[0]),
            'NaiveBayes': str(NaiveBayesClassifier.predict(user_data)[0]),
            'KNeighbors': str(KNeighborsClassifier.predict(user_data)[0])
        },
        'Accuracy': {
            'LinearSVC': str(75.32467532467533),
            'NaiveBayes': str(74.67532467532467),
            'KNeighbors': str(77.27272727272727)
        }
    }
    
```

Fig 6.8 - API to predict Diabetes

```

@app.route("/api/heart", methods=['GET', 'POST'])
def predict_heart():
    standard_scalar = GetStandardScalarForHeart()

    data = request.get_json(force=True)

    user = [{ 'age': data['age'],
              'sex': data['sex'],
              'cp': data['cp'],
              'trestbps': data['trestbps'],
              'chol': data['chol'],
              'fbs': data['fbs'],
              'restecg': data['restecg'],
              'thalach': data['thalach'],
              'exang': data['exang'],
              'oldpeak': data['oldPeak'],
              'slope': data['slope'],
              'ca': data['ca'],
              'thal': data['thal']
            }]

    user_data = pd.DataFrame(user)
    user_data = standard_scalar.transform(user_data)

    LinearSVCClassifier, NaiveBayesClassifier, KNeighborsClassifier = GetAllClassifiersForHeart()
    predict_result = {
        'Predictions': {
            'LinearSVC': str(LinearSVCClassifier.predict(user_data)[0]),
            'NaiveBayes': str(NaiveBayesClassifier.predict(user_data)[0]),
            'KNeighbors': str(KNeighborsClassifier.predict(user_data)[0])
        },
        'Accuracy': {
            'LinearSVC': str(86.88524590163934),
            'NaiveBayes': str(85.24590163934425),
            'KNeighbors': str(85.24590163934425)
        }
    }
    return make_response(jsonify(predict_result), 200)
    
```

Fig 6.9 - API to predict Heart Disease

CHAPTER 7

RESULTS AND DISCUSSION

The Prediction Engine provides an optimal performance with the right dataset and efficient training of the classifier models considering all aspects and a lot of learning from the previous experiences. The implemented Prediction Engine is capable of predicting the presence of Diabetes with an accuracy of,

- Linear SVM: 75.32467532467533
- Naïve Bayes: 74.67532467532467
- KNN: 77.27272727272727

And the presence of Heart Disease with an accuracy of,

- Linear SVM: 86.88524590163934
- Naïve Bayes: 85.24590163934425
- KNN: 85.24590163934425

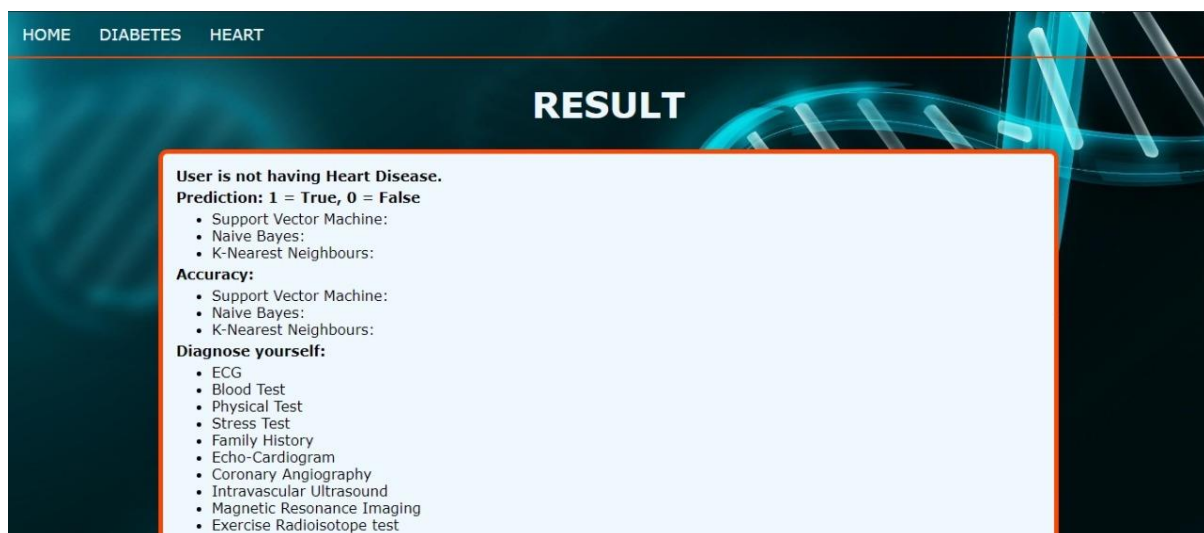


Fig 7.1 - Prediction Engine Result Page

The Front-End of the prediction engine was developed by making use of the Angular Framework (HTML, CSS and Typescript). The API calls and results were handled by importing the HTTP Module of the Angular Framework.

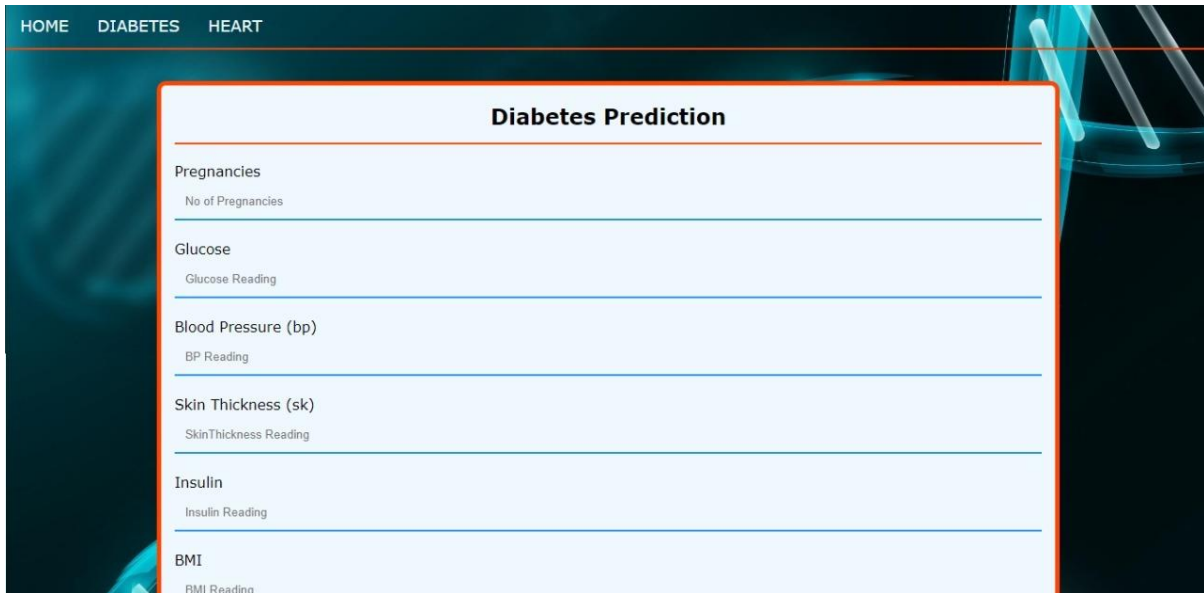


Fig 7.2 - Diabetes Form to collect User Data for Prediction



Fig 7.3 - Heart Disease Form to collect User Data for Prediction

CHAPTER 8

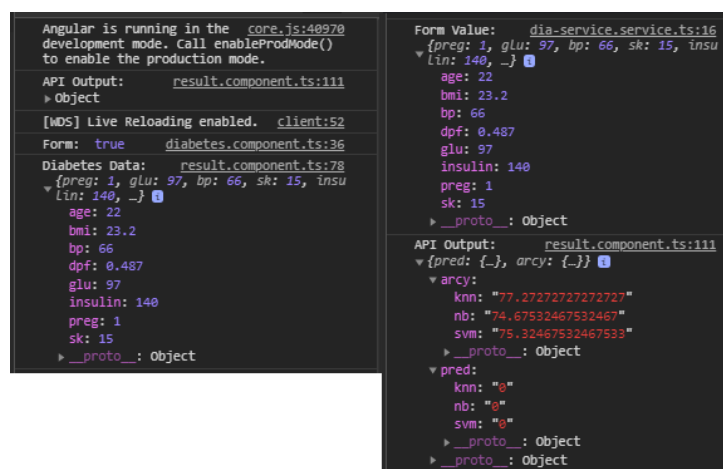
TESTING

The KNN, SVM and Naïve Bayes Models were trained using 80% of the dataset and were tested using the other 20% of the dataset. Overall, the models gave an average efficiency of 75% and 85% for diabetes and heart disease prediction respectively.

In concern with testing of API's and Front-End, postman and Angular HTTP Modules were used. Every time the user submits an input form, the input form object with the input values is logged on the console to validate whether the user has entered valid data. The user input form has validation checks enforced on each input field of the form to validate whether the user has entered correct data.

Before making an API call with the form input object of heart or diabetes form data, the data will be logged in the console to validate the information that goes in the POST Method of the API call, providing the developer information of what goes into the POST Method and is the User Input.

Every time an API call is made, the body of the GET Request returned by the API is parsed using JSON Module of Angular Framework and will be stored in an object which is again logged on the console to verify the expected behaviour.



```

Angular is running in the core.js:40970
development mode. Call enableProdMode()
to enable the production mode.

API output: result.component.ts:111
> Object

[MDS] Live Reloading enabled. client:52
Form: true diabetes.component.ts:36
Diabetes Data: result.component.ts:78
{preg: 1, glu: 97, bp: 66, sk: 15, insu
  Lin: 140, _}
  age: 22
  bmi: 23.2
  bp: 66
  dpf: 0.487
  glu: 97
  insulin: 140
  preg: 1
  sk: 15
  __proto__: Object

Form Value: dia-service.service.ts:16
{preg: 1, glu: 97, bp: 66, sk: 15, insu
  Lin: 140, _}
  age: 22
  bmi: 23.2
  bp: 66
  dpf: 0.487
  glu: 97
  insulin: 140
  preg: 1
  sk: 15
  __proto__: Object

API Output: result.component.ts:111
{pred: {}, arcy: {}}
  arcy:
    knn: "77.27272727272727"
    nb: "74.67532467532467"
    svm: "75.32467532467533"
    __proto__: Object
  pred:
    knn: "0"
    nb: "0"
    svm: "0"
    __proto__: Object
  __proto__: Object
  
```

Fig 8.1 - Form Validation and API Testing

The KNN, SVM and Naïve Bayes trained models were tested out by replicating user input data in the form of JSON objects in the python scripts and calling the trained models by providing these objects post processing. The models gave promising results and displayed the expecting behaviour.

```
In [43]: user1 = [{'Pregnancies' : 7,
                'Glucose' : 147,
                'BloodPressure' : 76,
                'SkinThickness' : 0,
                'Insulin' : 0,
                'BMI' : 39.4,
                'DiabetesPedigreeFunction' : 0.257,
                'Age' : 43}]

user2 = [{'Pregnancies' : 1,
          'Glucose' : 97,
          'BloodPressure' : 66,
          'SkinThickness' : 15,
          'Insulin' : 140,
          'BMI' : 23.2,
          'DiabetesPedigreeFunction' : 0.487,
          'Age' : 22}]

dataset_ = [{'Glucose' : 121.68676277850587,
             'BloodPressure' : 72.40518417462486,
             'SkinThickness' : 29.153419593345007,
             'Insulin' : 155.5482233502544,
             'BMI' : 32.457463672391
            }]

dataset_ = pd.DataFrame(dataset_)

#7,147,76,0,0,39.4,0.257,43,1 - yes
#1,97,66,15,140,23.2,0.487,22,0 - no

Z_test
Out[43]:


|   | Pregnancies | Glucose | BloodPressure | SkinThickness | Insulin    | BMI  | DiabetesPedigreeFunction | Age |
|---|-------------|---------|---------------|---------------|------------|------|--------------------------|-----|
| 0 | 7           | 147     | 76            | 29.15342      | 155.548223 | 39.4 | 0.257                    | 43  |



In [44]: Z_test=sc_X.transform(Z_test)

Out[44]: array([[ 0.98391937,  0.83630666,  0.31056364,  0.04730861,  0.00739544,
                  1.01684019, -0.63037788,  0.87809089]])

In [45]: knn_pred = knn.predict(Z_test)
svc_pred = svc.predict(Z_test)
nb_pred = nb.predict(Z_test)

print('KNN Accuracy: ',accuracy_knn*100)
print('KNN Result:', knn_pred[0])
print('SVM Accuracy: ',accuracy_svc*100)
print('SVM Result:', svc_pred[0])
print('NB Accuracy: ',accuracy_nb*100)
print('NB Result:', nb_pred[0])

KNN Accuracy: 77.27272727272727
KNN Result: 1
SVM Accuracy: 75.32467532467533
SVM Result: 1
NB Accuracy: 74.67532467532467
NB Result: 1
```

Fig 8.2 - Trained Models tested for Diabetes Prediction

```
In [11]: user1 = [{'age' : 58,
                 'sex' : 1,
                 'cp' : 2,
                 'trestbps' : 132,
                 'chol' : 224,
                 'fbs' : 0,
                 'restecg' : 0,
                 'thalach' : 173,
                 'exang' : 0,
                 'oldpeak' : 3.2,
                 'slope' : 2,
                 'ca' : 2,
                 'thal' : 3}]

user2 = [{'age' : 76,
          'sex' : 0,
          'cp' : 2,
          'trestbps' : 140,
          'chol' : 197,
          'fbs' : 0,
          'restecg' : 2,
          'thalach' : 116,
          'exang' : 0,
          'oldpeak' : 1.1,
          'slope' : 1,
          'ca' : 0,
          'thal' : 2}]

#58,1,2,132,224,0,0,173,0,3.2,2,2,3,0 - no
#76,0,2,140,197,0,2,116,0,1.1,1,0,2,1 - yes

Z_test = pd.DataFrame(user1)
Z_test=sc_X.transform(Z_test)

Out[11]: array([[ 0.35276583,  0.69617712,  0.98584243,  0.01137121, -0.4265324 ,
                  -0.41803981, -0.97936664,  1.00226448, -0.70929937,  1.79365833,
                  0.95577901,  1.23743687,  1.14190596]])

In [12]: knn_classifier.predict(Z_test)
svm_classifier.predict(Z_test)
nb_classifier.predict(Z_test)

print('KNN Accuracy: ',knn_accuracy*100)
print('KNN Result:', knn_pred[0])
print('SVM Accuracy: ',svm_accuracy*100)
print('SVM Result:', svm_pred[0])
print('NB Accuracy: ',nb_accuracy*100)
print('NB Result:', nb_pred[0])

KNN Accuracy: 85.24590163934425
KNN Result: 0
SVM Accuracy: 86.88524590163934
SVM Result: 0
NB Accuracy: 85.24590163934425
NB Result: 0

In [13]: import pickle
pickle.dump(knn_classifier, open('knn_hrt.pkl', 'wb'))
```

Fig 8.3 - Trained Models tested for Heart Disease Prediction

CHAPTER 9

CONCLUSION AND FUTURE SCOPE

9.1 Conclusion

We developed a Prediction Engine which enables the user to check whether he/she has diabetes or heart disease. The user interacts with the Prediction Engine by filling a form which holds the parameter set provided as an input to the trained models. The Prediction engine provides an optimal performance compared to other state of art approaches. The Prediction Engine makes use of three algorithms to predict the presence of a disease namely: Support Vector Machine (SVM), K-Nearest Neighbours (KNN) and Naïve Bayes. The reason to choose these three algorithms are:

- They are effective, if the training data is large.
- A single dataset can be provided as an input to all these 3 algorithms with minimal or no modification.
- A common scalar can be used to normalize the input provided to these 3 algorithms.

9.2 Future Work

- To enhance the functionality of the prediction engine providing the details of 5 nearest hospitals or medical facilities to the user input location.
- Provide a user account which allows the user to keep track of their medical test data and get suggestions or support to meet the right specialists or the tests to be taken
- Provide admin controls to upload, delete the dataset which will be used to train the model.

- Automate the process of training the model and extracting pickle files of the trained models which will be consumed by the API's to predict the disease.
- Mail the detailed report of the prediction engine results along with the information of 5 nearest medical facilities details having location and contact information.

REFERENCES

- [1] Priyanka Sonar, Prof. K. JayaMalini, “Diabetes Prediction Using Different Machine Learning Approaches”, *Proceedings of the Third International Conference on Computing Methodologies and Communication (ICCMC 2019) IEEE Xplore Part Number: CFP19K25-ART; ISBN: 978-1-5386-7808-4*
- [2] Samrat Kumar Dey, Ashraf Hossain, Md. Mahbubur Rahman, “Implementation of a Web Application to Predict Diabetes Disease: An Approach Using Machine Learning Algorithm”, *2018 21st International Conference of Computer and Information Technology (ICCIT)*
- [3] Deeraj Shetty, Kishor Rit, Sohail Shaikh, Nikita Patil, “Diabetes Disease Prediction Using Data Mining”, *2017 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS)*
- [4] Zhilbert Tafa, Nerxhivane Pervetica, Bertran Karahoda, “An Intelligent System for Diabetes Prediction”, *4thMediterranean Conference on Embedded Computing MECO – 2015 Budva, Montenegro*