A REPORT OF 04 WEEKS INDUSTRIAL TRAINING

At

ASPEXX Health Solution Pvt. Ltd.

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE

AWARD OF THE DEGREE OF

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE ENGINEERING

28 th Jun, 2021- 23 rd July, 2021

SUBMITTED BY:

NAME: Upendra Gupta

USN: 1AM18CS186

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

AMC Engineering College
18th K.M, Bannerghatta Main Road, Bangalore-560083
2020 - 21

CONTENTS

CERTIFICATION

This is to certify that this project report entitled “ **AI , ML & Data Science**” *by Upendra Gupta* submitted in partial fulfilment of the requirements for the internship (ASPEXX Health Solution Pvt. Ltd.), during the Four weeks of internship , is a bonafede record of work carried out under my guidance and supervision. I hereby declare that the work has been carried out under my supervision and has not been submitted elsewhere for any other purpose.

(Signature of CEO)
Ms. Shivani Mishra

(Signature of Director)

Managing Director

VISVESVARAYA TECHNOLOGICAL UNIVERSITY

CANDIDATE'S DECLARATION

I Upendra Gupta hereby declare that I have undertaken 04 weeks Training from ASPEXX Health Solution Pvt. Ltd. during a period from June 28, 2021- July 24, 2021 in partial fulfillment of requirements for the award of degree of B.E Computer Science & Engineering at Visvesvaraya Technological University. The work which is being presented in the training report submitted to the Department of Computer Science & Engineering at Faculty of Engineering & Technology, Bangalore is an authentic record of training work.

Name of Student: Upendra Gupta Roll No: 1AM18CS186

(Signature of Director)

Managing Director

ABSTRACT

Software training is one of the requirements to be fulfilled in order to obtain the Bachelor's degree in technology. Each student needs to do software training in a recognized company of their respective domain. The students are required to do training for a duration of 1 months which is intended for their exposure to the software industry. A well planned, properly executed and evaluated software training helps a lot in developing a professional attitude. It develops an awareness of software approach to problem solving, based on a broad understanding of processes. Besides, software training builds self confidence among students and lets students know technical knowledge and professionalism.

During internship from ASPEXX Health Solution Pvt. Ltd. most of the theoretical knowledge gained during the course of studies was put to test. Various efforts and processes involved in designing a component were studied and understood during the internship. In our internship, I undertook projects of **AI**.

The training gave me good experience from the view of implementing my theoretical knowledge in practical aspects. It gave me firsthand experience of working as an engineering professional. It helped me in improving my technical, interpersonal and communication skills, both oral and written. Overall, it is a great experience to have industrial training in such a reputed firm and I believe that it will help me in building a successful career.

ACKNOWLEDGMENT

Any effort becomes successful when there is the effect of synergy-the concept that two and two make more than four. This report has also the effect of synergy, without prejudice to my own contribution. I am appreciative of **ASPEXXHealth Solutions Pvt. Ltd** Company and those people who cooperate with me. It is my privilege that I had the opportunity to do an internship at **ASPEXX HealthSolutions Pvt. Ltd** Company, India. I would like to thank those all who either directly or indirectly contributed to this project.

- First, I express my deep gratitude to Bajarang Prasad Mishra and ShivaniMishra. of **ASPEXX Health Solutions Pvt. Ltd** who gave me the opportunity to allow me in this organization.

I would also like to express my excessive thanks to all members of **ASPEXX Health Solutions Pvt. Ltd** for their excellent support and proper guidance in completing my internship report.

- Upendra Gupta

<u>CHAPTER 1</u>

1.1 INTRODUCTION

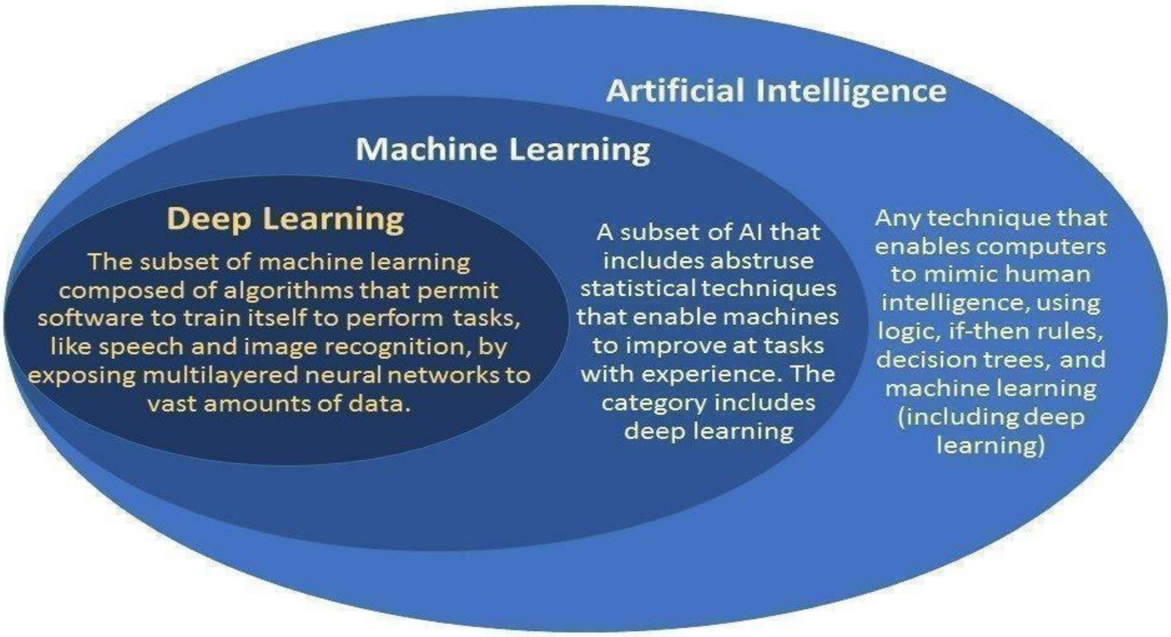
CUREYA (registered as Aspexx Health solutions Pvt Ltd , under MCA) is DPIIT recognized startup , registered under ' STARTUP INDIA SCHEME ' .CUREYA collaborated with stakeholders include - World Yoga associations , Flag bits technologies and many more . CUREYA primarily objective is ' HEALTH FOR ALL', by reducing the medical expenditure , eliminating the information asymmetry , promoting health awareness and achieving inclusive & holistic approach for healthcare treatments . The mission is to achieve the right to "Health for All" and improve the healthcare indicators by dissemination of health education that focuses on health promotion, health prevention, and self- medication. The objective is to eliminate the information asymmetry, language barrier, and emphasize to achieve global standards of healthcare delivery systems based on access , equity, affordability and quality, efficiency, sustainability.

Social Media Links –

1. Facebook- https://www.facebook.com/cureya7
2. 2- Instagram- https://www.instagram.com/cureya.in/
3. YouTube-https://www.youtube.com/channel/UCjsRwGm--mr1ADln5CB5Siw/videos
4. LinkedIn (Company)- https://www.linkedin.com/company/28749699 5- Website - www.cureya.in

Chapter 2

Difference between AI, ML, Deep Learning


Artificial Intelligence
Machine Learning
Deep Learning
The subset of machine learning composed of algorithms that permit software to train itself to perform tasks, like speech and image recognition, by exposing multilayered neural networks to vast amounts of data.
A subset of AI that includes abstruse statistical techniques that enable machines to improve at tasks with experience. The category includes deep learning
Any technique that enables computers to mimic human intelligence, using logic, if-then rules, decision trees, and machine learning (including deep learning)

Comparison between AI, ML, Deep Learning and DataScience.

What is Artificial Intelligence or AI?

Artificial Intelligence describes machines that can perform tasks resembling those of humans. So AI implies machines that artificially model human intelligence. AI systems help us manage, model, and analyze complex systems. It is the superset which has Machine Learning & Deep Learning as subset.

What is Machine Learning or ML?

Machine learning uses algorithms to parse data, learn from that data, and make informed decisions based on what it has learned.

What is Deep Learning or DL?

Deep learning structures algorithms in layers to create an “artificial neural network” that can learn and make intelligent decisions on its own. Deep learning is a subfield of machine learning. While both fall under the broad category of artificial intelligence, deep learning is what powers the most human-like artificial intelligence.

What is Data Science?

Data science is a broad field that spans the collection, management, analysis and interpretation of large amounts of data with a wide range of applications. It integrates all the terms above and summarizes or extract insights from data (exploratory data analysis) and make predictions from large datasets (predictive analytics).

Learn Python using Jupyter Notebook

The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. Uses include: data cleaning and transformation, numerical simulation, statistical modelling, data visualization, machine learning, and much more.

Jupyter Notebooks are a spin-off project from the IPython project, which used to have an IPython Notebook project itself. The name, Jupyter, comes from the core supported programming languages that it supports: Julia, Python, and R. Jupyter ships with the IPython kernel, which allows you to write your programs in Python, but there are currently over 100 other kernels that you can also use.

Chapter 3

Basic of Machine Learning Algorithm

Broadly, there are 3 types of Machine Learning Algorithms

1. Supervised Learning

How it works: This algorithm consists of a target / outcome variable (or dependent variable) which is to be predicted from a given set of predictors (independent variables). Using these sets of variables, we generate a function that maps inputs to desired outputs. The training process continues until the model achieves a desired level of accuracy on the training data.
Examples of Supervised Learning: Regression, Decision Tree, Random Forest, KNN, Logistic Regression etc.

2. Unsupervised Learning

How it works: In this algorithm, we do not have any target or outcome variable to predict / estimate. It is used for clustering populations in different groups, which is widely used for segmenting customers in different groups for specific intervention. Examples of Unsupervised Learning: Apriori algorithm, K-means.

3. Reinforcement Learning:

How it works: Using this algorithm, the machine is trained to make specific decisions. It works this way: the machine is exposed to an environment where it trains itself continually using trial and error. This machine learns from past experience and tries to capture the best possible knowledge to make accurate business decisions. Example of Reinforcement Learning: Markov Decision Process

Machine Learning Application

1. Image Recognition
2. Speech Recognition
3. Traffic prediction
4. Product recommendations
5. Self-driving cars
6. Email Spam and Malware Filtering
7. Virtual Personal Assistant
8. Online Fraud Detection
9. Stock Market trading
10. Medical Diagnosis

Machine Learning Model Approaches

A Taxonomy of Machine Learning Models

There is no simple way to classify machine learning algorithms. In this section, we present a taxonomy of machine learning models adapted from the book Machine Learning by Peter Flach. While the structure for classifying algorithms is based on the book, the explanation presented below is created by us.

For a given problem, the collection of all possible outcomes represents the sample space or instance space.

The basic idea for creating a taxonomy of algorithms is that we divide the instance space by using one of three ways:

- Using a Logical expression.
- Using the Geometry of the instance space.
- Using Probability to classify the instance space.

The outcome of the transformation of the instance space by a machine learning algorithm using the above techniques should be exhaustive (coverall possible outcomes) and mutually exclusive (non-overlapping).

Machine Learning Fundamentals

1. Machine Learning is an application of artificial intelligence where a computer/machine learns from the past experiences (input data) and makes future predictions. The performance of such a system should be at least human level.

2. Machine Learning Categories. Machine Learning is generally categorized into three types: Supervised Learning, Unsupervised Learning, Reinforcement learning.

3. The main aim of training the ML algorithm is to adjust the weights W to reduce the MAE or MSE.

4. Gradient descent Algorithm: There are three ways of doing gradient descent: Batch gradient descent: Uses all of the training instances to update the model parameters in each iteration. Mini-batch Gradient Descent: Instead of using all examples, Mini-batch Gradient Descent divides the training set into a smaller size called batch denoted by ‘b’.

   Thus, a mini-batch ‘b’ is used to update the model parameters in each iteration. Stochastic Gradient Descent (SGD): updates the parameters using only a single training instance in each iteration. The training instance is usually selected randomly. Stochastic gradient descent is often preferred to optimize cost functions when there are hundreds of thousands of training instances or more, as it will converge more quickly than batch gradient descent.

Machine Learning with Python

Python is the fifth most important language as well as the most popular language for Machine learning and data science. The following are the features of Python that makes it the preferred choice of language for data science −

Extensive set of packages:

Python has an extensive and powerful set of packages which are ready to be used in various domains. It also has packages like numpy,SciPy, pandas, scikit- learn etc. which are required for machine learning and data science.

Easy prototyping:

Another important feature of Python that makes it the choice of language for data science is the easy and fast prototyping. This feature is useful for developing new algorithms.

Collaboration feature:

The field of data science basically needs good collaboration andPython provides many useful tools that make this extremely useful.

One language for many domains:

A typical data science project includes various domains like data extraction, data manipulation, data analysis, feature extraction, modelling, evaluation, deployment and updating the solution. AsPython is a multi-purpose language, it allows the data scientist to address all these domains from a common platform.

Data Science: Gains using - SAS, R and Python

Data science is an interdisciplinary field that uses scientific methods, processes, algorithms and systems to extract knowledge and insights from data in various forms, both structured and unstructured, similar to data mining.

Data science is a "concept to unify statistics, data analysis, machine learning and their related methods' ' in order to "understand and analyze actual phenomena" with data. It employs techniques and theories drawn from many fields within the context of mathematics, statistics, information science and computer science.

SAS: SAS has been the undisputed market leader in the commercial analytics space. The software offers a huge array of statistical functions, has a good GUI(Enterprise Guide & Miner) for people to learn quickly and provides awesome technical support. However, it ends up being the most expensive option and is not always enriched with the latest statistical functions.

R: R is the Open-source counterpart of SAS, which has traditionally been used in academics and research. Because of its open-source nature, latest techniques get released quickly. There is a lot of documentation available over the internet and it is a very cost-effective option.

Python: With origination as an open-source scripting language, Python usage has grown over time. Today, it sports libraries (NumPy, SciPy and matplotlib) and functions for almost any statistical operation / model building you may want to do. Since the introduction of pandas, it has become very strong in operations on structured data.

Chapter 4

Statistics in Machine Learning

Statistics and machine learning are two very closely related fields. That statistical methods can be used to clean and prepare data ready for modelling. That statistical hypothesis tests and estimation statistics can aid in model selection and in presenting the skill and predictions from final models.

1. Statistics is a collection of tools that you can use to get answers to important questions about data. You can use descriptive statistical methods to transform raw observations into information that you can understand and share. ... Statistics is generally considered a prerequisite to the field of applied machine learning.

2. Machine Learning is an interdisciplinary field that uses statistics, probability, algorithms to learn from data and provide insights which can be used to build intelligent applications.

3. The major difference between machine learning and statistics is their purpose. Machine learning models are designed to make the most accurate predictions possible. Statistical models are designed for inference about the relationships between variables.

4. The field of statistics is the science of learning from data. Statistical knowledge helps you use the proper methods to collect the data, employ the correct analyses, and effectively present the results. Statistics allows you to understand a subject much more deeply.
5. Statistical learning theory was introduced in the late 1960s but until 1990s it was simply a problem of function estimation from a given

collection of data. ... Some more examples of the learning problems are: Predict whether a patient, hospitalized due to a heart attack, will have a second heart attack.

6. Linear regression is a technique, while machine learning is a goal that can be achieved through different means and techniques. So regression performance is measured by how close it fits an expected line/curve, while machine learning is measured by how good it can solve a certain problem, with whatever means necessary.

7. This is caused in part by the fact that Machine Learning has adopted many of Statistics' methods, but was never intended to replace statistics, or even to have a statistical basis originally “Machine Learning is statistics scaled up to big data” “The short answer is that there is no difference.

8. Most Important Methods For Statistical Data Analysis Mean. The arithmetic mean, more commonly known as “the average,” is the sum of a list of numbers divided by the number of items on the list. Standard Deviation. ...Regression .... Sample Size Determination…Hypothesis Testing.
9. The major difference between machine learning and statistics is their purpose. Machine learning models are designed to make the most accurate predictions possible. Statistical models are designed for inference about the relationships between variables.” ... You cannot do statistics unless you have data.

10. This is caused in part by the fact that Machine Learning has adopted many of Statistics' methods, but was never intended to replace statistics, or even to have a statistical basis originally

.... “Machine learning is statistics scaled up to big data” “The short answer is that there is no difference.

11. Statistics are used behind all the medical studies. Statistics help doctors keep track of where the baby should be in his/her mental development. Physician's also use statistics to examine the effectiveness of treatments. Statistics are very important for observation, analysis and mathematical prediction models.

12. Statistics is mostly used by doctors to explain risk to patients, accessing evidence summaries, interpreting screening test results and reading research publications

13. Statistical learning refers to the process of extracting this structure. A major question in language acquisition in the past few decades has been the extent to which infants use statistical learning mechanisms to acquire their native language.

14. Statistical learning theory is a framework for machine learning drawing from the fields of statistics and functional analysis. Statistical learning theory deals with the problem of finding a predictive function based on data.

Feature Engineering

1. Feature engineering is the process of transforming raw data into features that better represent the underlying problem to the predictive models, resulting in improved model accuracy on unseen data.

2. Feature engineering involves leveraging data mining techniques to extract features from raw data along with the use of domain knowledge. Feature engineering is useful to improve the performance of machine learning algorithms and is often considered as applied machine learning

3. Feature engineering is the process of using data's domain knowledge to create features that make machine learning algorithms work (Wikipedia). It's the act of extracting important features from raw data and transforming them into formats that are suitable for machine learning.

4. Feature Selection: Select a subset of input features from the dataset. Unsupervised: Do not use the target variable (e.g. remove redundant variables). Correlation. Supervised: Use the target variable (e.g. remove irrelevant variables). Wrapper: Search for well-performing subsets of features. RFE.

5. Feature engineering creates features from the existing raw data in order to increment the predictive power of the machine learning algorithms. Generally, the feature engineering process is applied to generate additional features from the raw data.

6. Engineering and selecting the correct features for a model will not only significantly improve its predictive power, but will also offer the flexibility to use less complex models that are faster to run and more easily understood.

7. The most common techniques of feature scaling are Normalization and Standardization. Normalization is used when we want to bound our values between two numbers, typically, between [0,1] or [-1,1]. While Standardization transforms the data to have zero mean and a variance of 1, they make our data unitless.

Machine Learning Pipelines

1. Machine learning pipeline. One definition of a machine learning pipeline is a means of automating the machine learning workflow by enabling data to be transformed and correlated into a model that can then be analyzed to achieve outputs. This type of ML pipeline makes the process of inputting data into the ML model fully automated.

2. Create the resources required to run an ML pipeline. Set up a datastore used to access the data needed in the pipeline steps. Configure a Dataset object to point to persistent data that lives in, or is accessible in, a datastore. Set up the compute targets on which your pipeline steps will run.

3. Data collection. Funnelling incoming data into a data store is the first step of any ML workflow. The key point is that data is persisted without undertaking any transformation at all, to allow us to have an immutable record of the original dataset.

4. A Reserve Component category designation that identifies untrained officers and enlisted personnel who have not completed initial active duty for training of 12 weeks or its equivalent. See also nondeployable account. Dictionary of Military and Associated Terms.
   US Department of Defense 2005.

5. One definition of a machine learning pipeline is a means of automating the machine learning workflow by enabling data to be transformed and correlated into a model that can then be analyzed to achieve outputs. This type of ML pipeline makes the process of inputting data into the ML model fully automated.

6. A machine learning pipeline is used to help automate machine learning workflows. They operate by enabling a sequence of data to be transformed and correlated together in a model that can be tested and evaluated to achieve an outcome, whether positive or negative.

7. Scikit-learn's pipeline class is a useful tool for encapsulating multiple different transformers alongside an estimator into one object, so that you only have to call your important methods once (fit() , predict() , etc).

What is PyTorch

1. PyTorch is an open-source machine learning library based on the Torch library, used for applications such as computer vision and natural language processing.

2. As you might be aware, PyTorch is an open source machine learning library used primarily for applications such as computer vision and natural language processing. PyTorch is a strong player in the field of deep learning and artificial intelligence, and it can be considered primarily as a research-first library.

3. So, both TensorFlow and PyTorch provide useful abstractions to reduce amounts of boilerplate code and speed up model development. The main difference between them is that PyTorch may feel more “pythonic” and has an object-oriented approach while TensorFlow has several options from which you may choose.

4. PyTorch is an open-source machine learning library based on the Torch library, used for applications such as computer vision and natural language processing, primarily developed by Facebook's AI Research lab (FAIR).

Chapter 5

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.

In the world of Big Data, data visualization tools and technologies are essential to analyze massive amounts of information and make data-driven decisions.

Data visualization is another form of visual art that grabs our interest and keeps our eyes on the message. When we see a chart, we quickly see trends and outliers. If we can see something, we internalize it quickly. It’s storytelling with a purpose. If you’ve ever stared at a massive spreadsheet of data and couldn’t see a trend, you know how much more effective a visualization can be.

Common general types of data visualization:

- Charts
- Tables
- Graphs
- Maps
- Infographics
- Dashboards

More specific examples of methods to visualize data:

- Area Chart
- Bar Chart
- Box-and-whisker Plots
- Bubble Cloud
- Bullet Graph
- Cartogram

- Circle View
- Dot Distribution Map
- Gantt Chart
- Heat Map
- Highlight Table
- Histogram
- Matrix
- Network
- Polar Area
- Radial Tree
- Scatter Plot (2D or 3D)
- Streamgraph
- Text Tables
- Timeline
- Treemap
- Wedge Stack Graph
- Word Cloud
- And any mix-and-match combination in a dashboard!

Interactive data visualization enables direct actions on a graphical plot to change elements and link between multiple plots.[33]

Interactive data visualization has been a pursuit of statisticians since the late 1960s. Examples of the developments can be found on the American Statistical Association video lending library.[34] Common interactions include:

-

- **Brushing**: works by using the mouse to control a paintbrush, directly changing the color or glyph of elements of a plot. The paintbrush is sometimes a pointer and sometimes works by drawing an outline of sorts around points; the outline is sometimes irregularly shaped, like a lasso. Brushing is most commonly used when multiple plots are visible and some linking mechanism exists between the plots. There are several different conceptual models for brushing and a number of common linking mechanisms. Brushing scatterplots can be a transient operation, in which points in the active plot only retain their new characteristics while they are enclosed or intersected by the brush, or it can be a persistent operation, so that points retain their new appearance after the brush has been moved away. Transient brushing is usually chosen for linked brushing, as we have just described.

- **Painting**: Persistent brushing is useful when we want to group the points into clusters and then proceed to use other operations, such as the tour, to compare the groups. It is becoming common terminology to call the persistent operation painting,

- **Identification**: which could also be called labeling or label brushing, is another plot manipulation that can be linked. Bringing the cursor near a point or edge in a scatterplot, or a bar in a barchart, causes a label to appear that identifies the plot element. It is widely available in many interactive graphics, and is sometimes called mouseover.

- **Scaling**: maps the data onto the window, and changes in the area of the. mapping function help us learn different things from the same plot. Scaling is commonly used to zoom in on crowded regions of a scatterplot, and it can also be used to change the aspect ratio of a plot, to reveal different features of the data.

Data Mapping

Data mapping is a way to organize various bits of data into a manageable and easy-to-understand system. This system matches data fields with target fields while in storage. Simply put, not all data goes by the same organizational standards. They may refer to a phone number in as many different ways as you can think of. Data mapping recognizes phone numbers for what they are and puts them all in the same field rather than having them drift around by other names.

With this technique, we're able to take the organized data and put a bigger picture together. You can find out where most of your target audience lives, learn what sorts of things they have in common and even figure out a few controversies that you shouldn't touch on.Armed with this information, your business can make smarter decisions and spend less money while spinning your products and services to your audience.

Data Mapping and Machine Learning

The earlier example of recognizing phone numbers has a lot to do with something called unification and data cleaning. These processes are often powered by machine learning, which is not to be confused with artificial intelligence.

Machine learning uses patterns and inference to offer predictions rather than perform a single task, which is more of a subset of AI technology than anything. In the earlier example, machine learning is used to recognize a phone number and assign it to its proper category for organizational purposes.

Machine learning goes a step beyond just recognizing phone numbers though. The technology can recognize errors like missing values or typos and group information from the same source together.

That's what data cleaning and unification really means — to clean up all of the data without any human input and present the information in its most perfect and precise form. This process saves time and is also more effective in regard to how correct the information will be.

The data can then be displayed in almost any way a person or company needs to see it. For instance, geospatial data is one route machine learning can automatically take and create without input. Geospatial data is basically translating data into a map and plotting out physical locations and routes that your target audience takes every day. This technique can provide a unique aid to your next advertising campaign.

Why Machine Learning Is Important to Data Mapping

Machine learning allows data mapping to be more precise. Without that technology, data mapping would be either very rudimentary or have to be done completely manually. Assuming we go the rudimentary route, a simple spreadsheet would be able to take information and plug into its best guess of a proper category. Typos wouldn't be fixed, missing values would remain missing and some information would just be scattered in random places.

Trying to complete data mapping manually would be worse. For one, a person would never be able to keep up with the flow of information, not to mention the backlog of information already hiding and in need of sorting in the Internet of Things. Assuming someone could keep up with the flow, there would still be errors as the sheer amount of data would lead to the human being unable to notice connections like a machine could.

Why Data Mapping Is Important

The use of data is an extremely important part of modern-day marketing. Knowing the best possible place and time to reach customers will allow you to target your audience more efficiently.Even large industries that can afford to splay their names in all possible media outlets use data mapping to save money and appear more loyal to their customer base.Big or small, you can use this information and get ahead of everyone else vying for your customers' attention. The competition is dense these days, so getting ahead of the curve and staying ahead is an art everyone is trying to perfect. Data mapping can help you get there as early as possible.

Uses of Data Map

Population Distribution

According to demographic data such as age, gender, income, education level, etc., analyze and classify customers in different regions or communities on the map. Data can help us figure out their lifestyle, interests and shopping habits.

Market Capacity Forecast

Analyze the resource investments, sales revenue, and product sales of each outlet on the map, and predict the capacity of the entire market, so that the resources can be scientifically allocated to the region with the greatest market potential.

Basics of Neural Networks

Neural networks, in the world of finance, assist in the development of processes such as time-series forecasting, algorithmic trading, securities classification, credit risk modeling and constructing proprietary indicators and price derivatives.

A neural network works similarly to the human brain's neural network. A "neuron" in a neural network is a mathematical function that collects and classifies information according to a specific architecture. The network bears a strong resemblance to statistical methods such as curve fitting and regression analysis.

A neural network contains layers of interconnected nodes. Each node is a perceptron and is similar to a multiple linear regression. The perceptron feeds the signal produced by a multiple linear regression into an activation function that may be nonlinear.

In a multi-layered perceptron (MLP), perceptrons are arranged in interconnected layers. The input layer collects input patterns. The output layer has classifications or output signals to which input patterns may map. For instance, the patterns may comprise a list of quantities for technical indicators about a security; potential outputs could be "buy," "hold" or "sell."

Hidden layers fine-tune the input weightings until the neural network's margin of error is minimal. It is hypothesized that hidden layers extrapolate salient features in the input data that have predictive power regarding the outputs. This describes feature extraction, which accomplishes a utility similar to statistical techniques such as principal component analysis.

Application of Neural Networks

Neural networks are broadly used, with applications for financial operations, enterprise planning, trading, business analytics and product maintenance. Neural networks have also gained widespread adoption in business applications such as forecasting and marketing research solutions, fraud detection and risk assessment.

A neural network evaluates price data and unearths opportunities for making trade decisions based on the data analysis. The networks can distinguish subtle nonlinear interdependencies and patterns other methods of technical analysis cannot. According to research, the accuracy of neural networks in making price predictions for stocks differs. Some models predict the correct stock prices 50 to 60 percent of the time while others are accurate in 70 percent of all instances. Some have posited that a 10 percent improvement in efficiency is all an investor can ask for from a neural network.

There will always be data sets and task classes that are better analyzed by using previously developed algorithms. It is not so much the algorithm that matters; it is the well-prepared input data on the targeted indicator that ultimately determines the level of success of a neural network.

Fuzzy Logic

Fuzzy logic is based on the observation that people make decisions based on imprecise and non-numerical information. Fuzzy models or sets are mathematical means of representing vagueness and imprecise information (hence the term fuzzy). These models have the capability of recognising, representing, manipulating, interpreting, and utilising data and information that are vague and lack certainty.[5]

Fuzzy logic has been applied to many fields, from control theory to artificial intelligence.

Fuzzification is the process of assigning the numerical input of a system to fuzzy sets with some degree of membership. This degree of membership may be anywhere within the interval [0,1]. If it is 0 then the value does not belong to the given fuzzy set, and if it is 1 then the value completely belongs within the fuzzy set. Any value between 0 and 1 represents the degree of uncertainty that the value belongs in the set. These fuzzy sets are typically described by words, and so by assigning the system input to fuzzy sets, we can reason with it in a linguistically natural manner.

For example, in the image below the meanings of the expressions cold, warm, and hot are represented by functions mapping a temperature scale. A point on that scale has three "truth values"—one for each of the three functions. The vertical line in the image represents a particular temperature that the three arrows (truth values) gauge. Since the red arrow points to zero, this temperature may be interpreted as "not hot"; i.e. this temperature has zero membership in the fuzzy set "hot". The orange arrow (pointing at 0.2) may describe it as "slightly warm" and the blue arrow (pointing at 0.8) "fairly cold". Therefore, this temperature has 0.2 membership in the fuzzy set "warm" and 0.8 membership in the fuzzy set "cold". The degree of membership assigned for each fuzzy set is the result of fuzzification.

Natural Language Processing

Natural language processing (NLP) is a branch of artificial intelligence that helps computers understand, interpret and manipulate human language. NLP draws from many disciplines, including computer science and computational linguistics, in its pursuit to fill the gap between human communication and computer understanding.

Natural language processing helps computers communicate with humans in their own language and scales other language-related tasks. For example, NLP makes it possible for computers to read text, hear speech, interpret it, measure sentiment and determine which parts are important.

Today's machines can analyze more language-based data than humans, without fatigue and in a consistent, unbiased way. Considering the staggering amount of unstructured data that's generated every day, from medical records to social media, automation will be critical to fully analyze text and speech data efficiently.

Structuring a highly unstructured data source

Human language is astoundingly complex and diverse. We express ourselves in infinite ways, both verbally and in writing. Not only are there hundreds of languages and dialects, but within each language is a unique set of grammar and syntax rules, terms and slang. When we write, we often misspell or abbreviate words, or omit punctuation. When we speak, we have regional accents, and we mumble, stutter and borrow terms from other languages.

While supervised and unsupervised learning, and specifically deep learning, are now widely used for modeling human language, there's also a need for syntactic and semantic understanding and domain expertise that are not necessarily present in these machine learning approaches. NLP is important because it helps resolve ambiguity in language and adds useful numeric structure to the data for many downstream applications, such as speech recognition or text analytics.

Genetic Algorithms

Nature has always been a great source of inspiration to all mankind. Genetic Algorithms (GAs) are search based algorithms based on the concepts of natural selection and genetics. GAs are a subset of a much larger branch of computation known as Evolutionary Computation.

GAs were developed by John Holland and his students and colleagues at the University of Michigan, most notably David E. Goldberg and has since been tried on various optimization problems with a high degree of success.

In GAs, we have a pool or a population of possible solutions to the given problem. These solutions then undergo recombination and mutation (like in natural genetics), producing new children, and the process is repeated over various generations. Each individual (or candidate solution) is assigned a fitness value (based on its objective function value) and the fitter individuals are given a higher chance to mate and yield more "fitter" individuals. This is in line with the Darwinian Theory of "Survival of the Fittest".

In this way we keep "evolving" better individuals or solutions over generations, till we reach a stopping criterion.

Genetic Algorithms are sufficiently randomized in nature, but they perform much better than random local search (in which we just try various random solutions, keeping track of the best so far), as they exploit historical information as well.

Advantages of GAs

GAs have various advantages which have made them immensely popular. These include −

- Does not require any derivative information (which may not be available for many real-world problems).
- Is faster and more efficient as compared to the traditional methods.
- Has very good parallel capabilities.
- Optimizes both continuous and discrete functions and also multi-objective problems.
- Provides a list of "good" solutions and not just a single solution.
- Always gets an answer to the problem, which gets better over time.
- Useful when the search space is very large and there are a large number of parameters involved.

Limitations of GAs

Like any technique, GAs also suffer from a few limitations. These include −

- GAs are not suited for all problems, especially problems which are simple and for which derivative information is available.

- Fitness value is calculated repeatedly which might be computationally expensive for some problems.
- Being stochastic, there are no guarantees on the optimality or the quality of the solution.
- If not implemented properly, the GA may not converge to the optimal solution.

Project

1. Introduction:

In today's scenario, the whole world is suffering from a pandemic. It’s not safe for anyone to go to hospital. We want to provide a platform where when a person enters his/her name and the 5 symptoms a person is observing in his/herself example like a person enters symptoms like losing weight , desiness, extra urination , thirsty, blurry vision then the person may be suffering from Diabetes. Further whole will be saved in the database whenever a person visits again the database can help to predict the disease.

2. Purpose and Scope:

The main aim of the project is to provide the platform to predict the disease a person(user) may be suffering on the basis of the various symptoms. The user can enter the symptoms and get the name of the disease.

3.Problem Statement:

The Problem statement for the project is when a new patient enters his/ her name and 5 symptoms, then our model should be capable of determining the disease the person is suffering from. Further all the details should be saved in the database.

Example:

If the person enter name as XYZ and symptoms as loss of weight, blurry vision, Urinate(pee) a lot often at night, mostly tiered, drink a lot of water (mostly thirsty), After going through all the examination and calculations the model will predict, that the person is suffering from diabetes. Further all the details will be saved in the database. So as to maintain a record if the save person suffers any disease we have all the records that can help us.

4. Project Analysis:

1. Review of Literature:

The main purpose of the scheme is to build the language gap between the user and health providers by giving immediate replies to the Questions asked by the user. Today's people are more likely addicted to the internet but they are not concerned about their personal health. They avoid going to hospital for small problems which may become a major disease in future. Establishing question answer forums is becoming a simple way to answer those queries rather than browsing through the list of potentially relevant documents from the web. Many of the existing systems have some limitations such as there is no instant response given to the patients they have to wait for experts to acknowledge for a long time. Some of the processes may charge an amount to perform live chat or telephony communication with doctors online.

The aim of this system is to replicate a person's discussion.

2. Project Timeline: Timeline provided was from June 28 2021 - July 24 2021

3. Dataset Details:

Dataset for this project was collected from a study of university of Columbia performed at New York Presbyterian Hospital during 2004. Link of dataset is given below.

http://people.dbmi.columbia.edu/~friedma/Projects/DiseaseSymptomKB/index.html

5. Methodology Used:

Using the method based approach, the project's core problem modules were divided into four distinct sections for identifying/ monitoring the appropriate input values to determine the end result of our project as a whole. These functions were separate testing models that we trained and tuned to determine the best result to determine and work on our new input data for real world problems. The core objective of using this kind of methodology is to deprecate any kind of complexity and keep the workflow as simplistic as possible. This way each model can stand on its own without any influence from others

These four methods/models are →

- NaiveBayes()
- KNN()
- randomforest()
- DecisionTree()

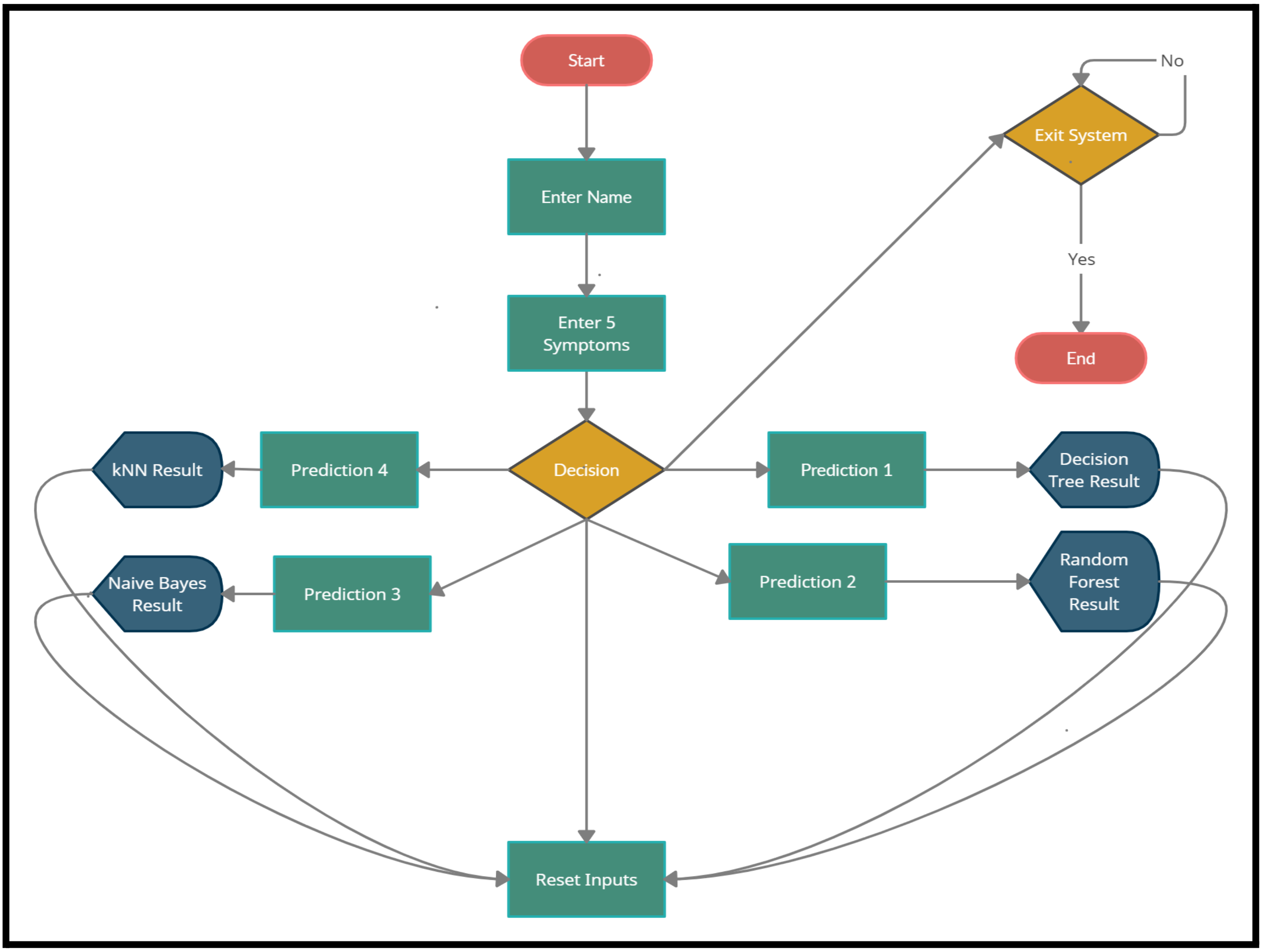
Apart from the the four major model functions and number of util functions are added as a means to increase code reusability, robustness, and overall efficiency of the project as a whole. These methods help in various visualizations and work as simple util functionalities like dialog confirmations or error message display boxes.

These methods are →

- scatter plot(disea)
- plotScatterMatrix(df1, plotSize, textSize)
- plotPerColumnDistribution(df1, nGraphShown, nGraphPerRow)
- Reset()
- help()
- about()
- quitApp()

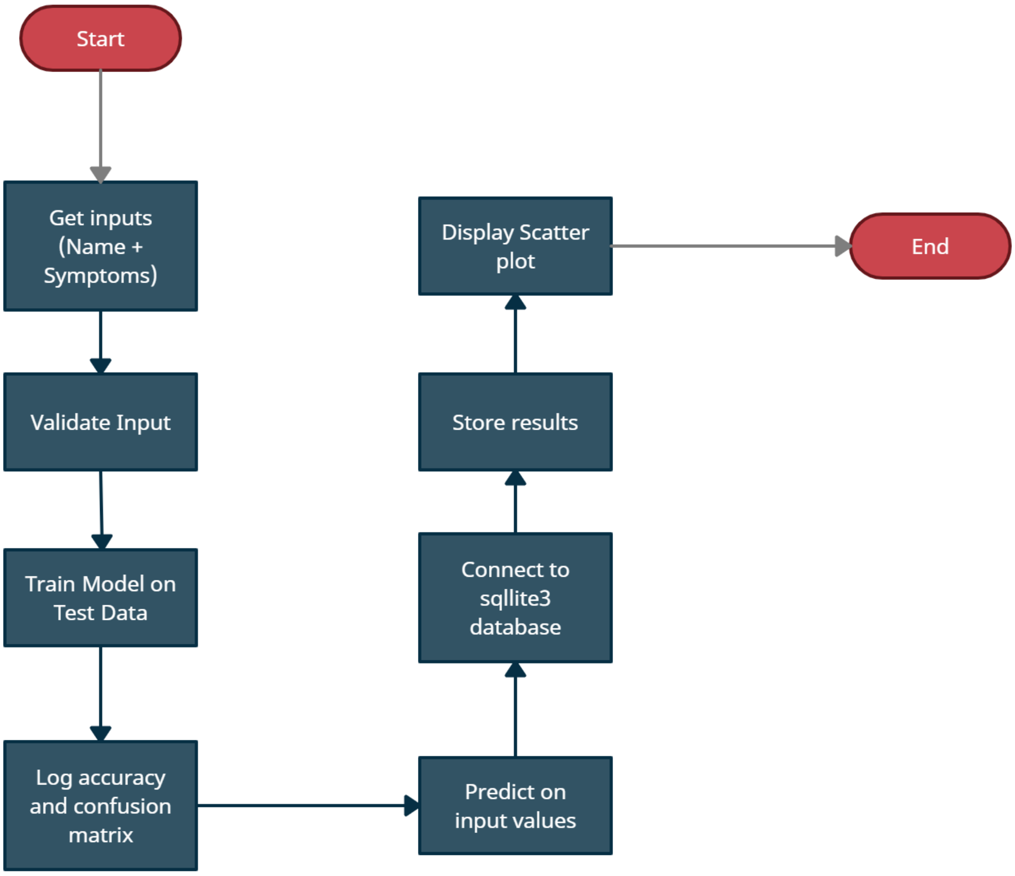
6. Project Design:

1. Block Diagram:



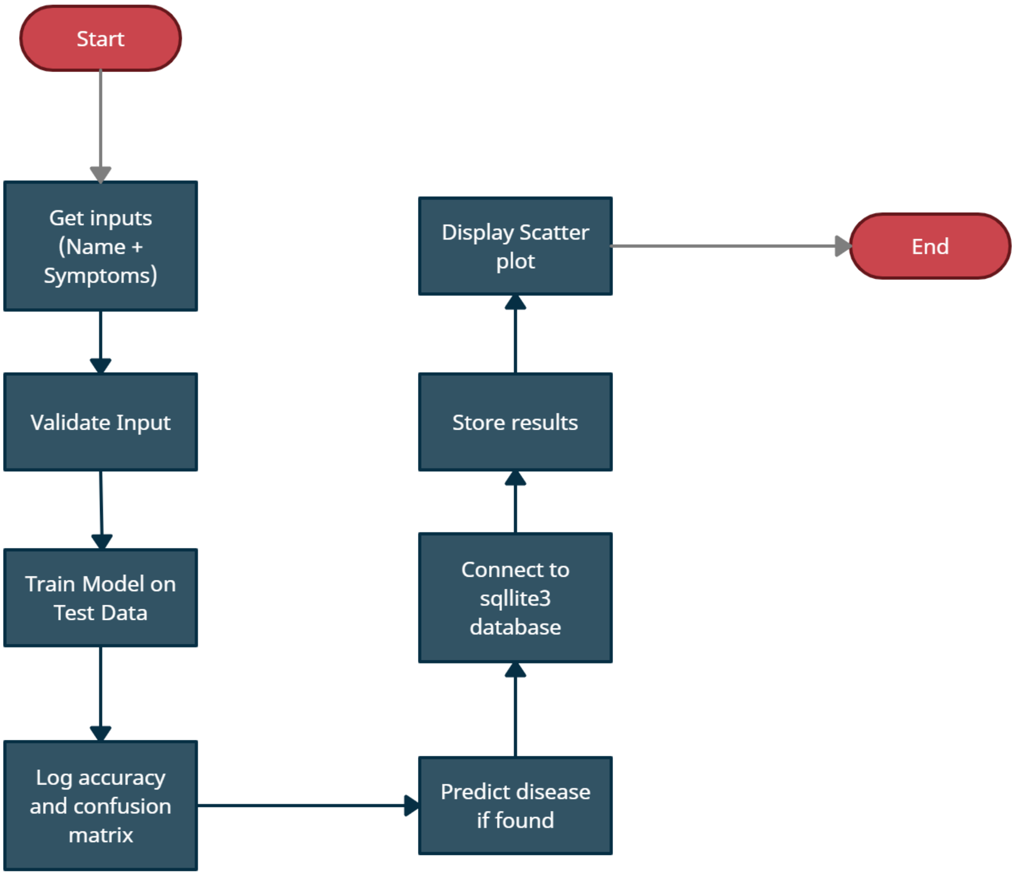
2. Method 1 Diagram:

Decision Tree Algorithm(DecisionTree():-)



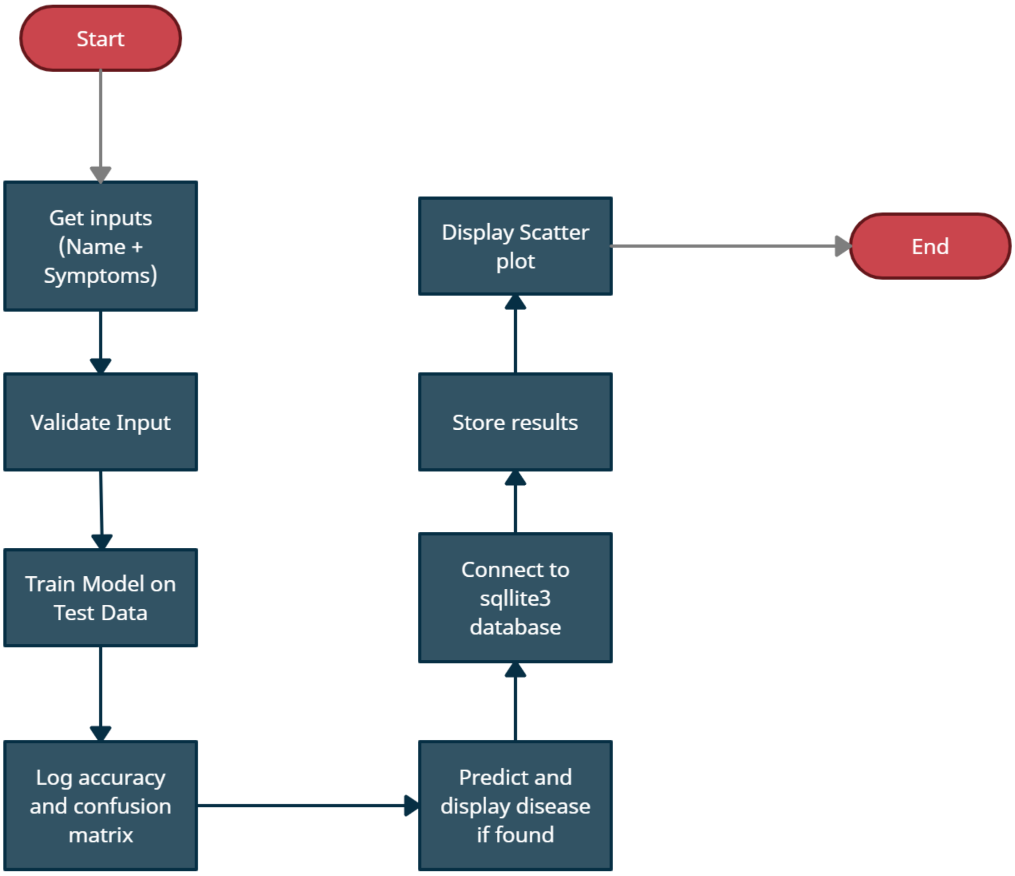
3. Method 2 Diagram:

Random Forest Algorithm(randomforest():-)



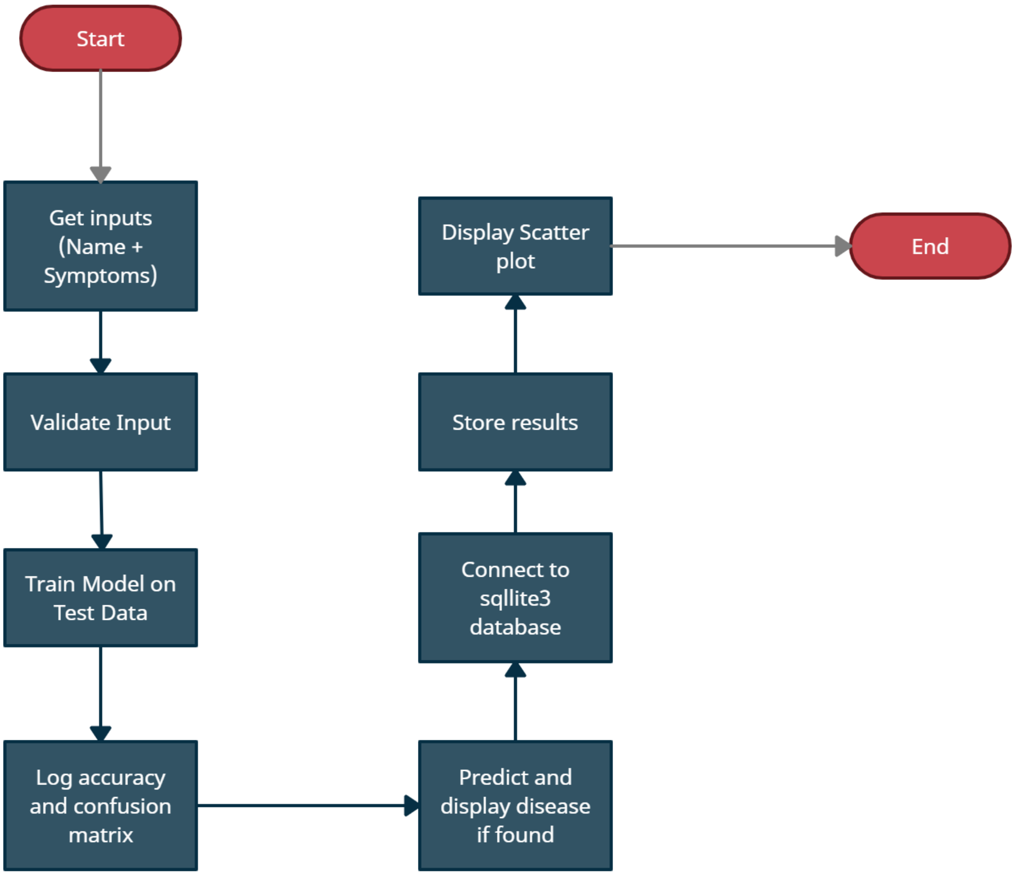
4. Method 3 Diagram:

KNearestNeighbour Algorithm(KNN():-)


Start
Get inputs (Name + Symptoms)
Validate Input
Train Model on Test Data
Log accuracy and confusion matrix
Predict and display disease if found
Connect to sqllite3 database
Store results
Display Scatter plot
End

5. Method 4 Diagram:

Naive Bayes Algorithm(NaiveBayes():-)


Start
Get inputs (Name + Symptoms)
Validate Input
Train Model on Test Data
Log accuracy and confusion matrix
Predict and display disease if found
Connect to sqllite3 database
Store results
Display Scatter plot
End

7. Implementation:

We set out to create a system which can predict disease on the basis of symptoms given to it. Such a system can decrease the rush at OPDs of hospitals and reduce the workload on medical staff. We were successful in creating such a system and used 4 different algorithms to do so. On an average we achieved accuracy of ~94%. Such a system can be largely reliable to do the job. Creating this system we also added a way to store the data entered by the user in the database which can be used in future to help in creating a better version of such a system. Our system also has an easy to use interface. It also has various visual representations of data collected and results achieved.

Work Division

1 **Upendra Gupta** – Implementation of ML and UI for project.

2 **Aneesha Sengupta** – Dataset and Basic Preprocessing and ML.

3 **Prateek Singh** – Choosing Best algorithm to implement.

4 **Vasvi Agarwal** – Coding and Report Preparation.

5 **Nikita Verma** – Enhancement of Report

6 **Madhuri Kumari** – Dataset cleaning

7 **Earavelly Sriharshitha** – Enhancement of PPT

8. Coding:

```
   # Disease Prediction based on Symptoms
#Importing Libraries
from mpl_toolkits.mplot3d import Axes3D #For visualization of data
from sklearn.preprocessing import StandardScaler #For algorithms
import matplotlib.pyplot as plt #For plotting graphs
from tkinter import * #For Gui
from tkinter import messagebox #For msg box
from tkinter.messagebox import showinfo #For msg box
import numpy as np #For scientific calculation
import pandas as pd #For data analysis
import os #For working with directory
#List of the symptoms is listed here in list l1.

l1=['abdominal_pain', 'abnormal_menstruation', 'acidity', 'acute_liver_failure',
'altered_sensorium', 'anxiety', 'back_pain', 'belly_pain', 'blackheads',
'bladder_discomfort', 'blister', 'blood_in_sputum', 'bloody_stool',
'blurred_and_distorted_vision', 'breathlessness', 'brittle_nails', 'bruising',
'burning_micturition', 'chest_pain', 'chills', 'cold_hands_and_feets', 'coma',
'congestion', 'constipation', 'continuous_feel_of_urine', 'continuous_sneezing',
'cough', 'cramps', 'dark_urine', 'dehydration', 'depression', 'diarrhoea', 'dischromic
_patches', 'distention_of_abdomen', 'dizziness', 'drying_and_tingling_lips',
'enlarged_thyroid', 'excessive_hunger', 'extra_marital_contacts', 'family_history',
'fast_heart_rate', 'fatigue', 'fluid_overload', 'fluid_overload', 'foul_smell_of urine',
'headache', 'high_fever', 'hip_joint_pain', 'history_of_alcohol_consumption',
'increased_appetite', 'indigestion', 'inflammatory_nails', 'internal_itching',
'irregular_sugar_level', 'irritability', 'irritation_in_anus', 'itching', 'joint_pain',
'knee_pain', 'lack_of_concentration', 'lethargy', 'loss_of_appetite',
'loss_of_balance', 'loss_of_smell', 'malaise', 'mild_fever', 'mood_swings',
'movement_stiffness', 'mucoid_sputum', 'muscle_pain', 'muscle_wasting',
'muscle_weakness', 'nausea', 'neck_pain', 'nodal_skin_eruptions', 'obesity',
'pain_behind_the_eyes', 'pain_during_bowel_movements', 'pain_in_anal_region',
'painful_walking', 'palpitations', 'passage_of_gases', 'patches_in_throat', 'phlegm',
'polyuria', 'prognosis', 'prominent_veins_on_calf', 'puffy_face_and_eyes',
'pus_filled_pimples', 'receiving_blood_transfusion',
'receiving_unsterile_injections', 'red_sore_around_nose', 'red_spots_over_body',
'redness_of_eyes', 'restlessness', 'runny_nose', 'rusty_sputum', 'scurring', 'shivering',
'silver_like_dusting', 'sinus_pressure', 'skin_peeling', 'skin_rash', 'slurred_speech',
'small_dents_in_nails', 'spinning_movements', 'spotting_ urination', 'stiff_neck',
'stomach_bleeding', 'stomach_pain', 'sunken_eyes', 'sweating',
'swelled_lymph_nodes', 'swelling_joints', 'swelling_of_stomach',
'swollen_blood_vessels', 'swollen_extremeties', 'swollen_legs', 'throat_irritation',
'toxic_look_(typhos)', 'ulcers_on_tongue', 'unsteadiness', 'visual_disturbances',
'vomiting', 'watering_from_eyes', 'weakness_in_limbs',
'weakness_of_one_body_side', 'weight_gain', 'weight_loss', 'yellow_crust_ooze',
'yellow_urine', 'yellowing_of_eyes', 'yellowish_skin']

#List of Diseases is listed in list disease.

disease=['Fungal infection', 'Allergy', 'GERD', 'Chronic cholestasis',
       'Drug Reaction', 'Peptic ulcer diseae', 'AIDS', 'Diabetes ',
       'Gastroenteritis', 'Bronchial Asthma', 'Hypertension ', 'Migraine',
       'Cervical spondylosis', 'Paralysis (brain hemorrhage)', 'Jaundice',
       'Malaria', 'Chicken pox', 'Dengue', 'Typhoid', 'hepatitis A',
       'Hepatitis B', 'Hepatitis C', 'Hepatitis D', 'Hepatitis E',
       'Alcoholic hepatitis', 'Tuberculosis', 'Common Cold', 'Pneumonia',
       'Dimorphic hemmorhoids(piles)', 'Heart attack', 'Varicose veins',
       'Hypothyroidism', 'Hyperthyroidism', 'Hypoglycemia',
       'Osteoarthristis', 'Arthritis',
       '(vertigo) Paroymsal  Positional Vertigo', 'Acne',
       'Urinary tract infection', 'Psoriasis', 'Impetigo']
l2=[]
for i in range(0,len(l1)):
   l2.append(0)
print(l2)
#Reading the training .csv file
df=pd.read_csv("training.csv")
DF= pd.read_csv('training.csv', index_col='prognosis')
#Replace the values in the imported file by pandas by the inbuilt function replace
in pandas.
```

```
df.replace({'prognosis':{'Fungal infection':0,'Allergy':1,'GERD':2,'Chronic
cholestasis':3,'Drug Reaction':4,
    'Peptic ulcer diseae':5,'AIDS':6,'Diabetes ':7,'Gastroenteritis':8,'Bronchial
Asthma':9,'Hypertension ':10,
    'Migraine':11,'Cervical spondylosis':12,
    'Paralysis (brain hemorrhage)':13,'Jaundice':14,'Malaria':15,'Chicken
pox':16,'Dengue':17,'Typhoid':18,'hepatitis A':19,
    'Hepatitis B':20,'Hepatitis C':21,'Hepatitis D':22,'Hepatitis E':23,'Alcoholic
hepatitis':24,'Tuberculosis':25,
    'Common Cold':26,'Pneumonia':27,'Dimorphic hemmorhoids(piles)':28,'Heart
attack':29,'Varicose veins':30,'Hypothyroidism':31,
    'Hyperthyroidism':32,'Hypoglycemia':33,'Osteoarthristis':34,'Arthritis':35,
    '(vertigo) Paroymsal  Positional Vertigo':36,'Acne':37,'Urinary tract
infection':38,'Psoriasis':39,
    'Impetigo':40}},inplace=True)

#df.head() Printing the head of data
DF.head()
# Distribution graphs (histogram/bar graph) of column data[For visualization of
dataset]
def plotPerColumnDistribution(df1, nGraphShown, nGraphPerRow):
    nunique = df1.nunique()
    df1 = df1[[col for col in df if nunique[col] > 1 and nunique[col] < 50]] # For
displaying purposes, pick columns that have between 1 and 50 unique values
    nRow, nCol = df1.shape
    columnNames = list(df1)
    nGraphRow = (nCol + nGraphPerRow - 1) / nGraphPerRow
    plt.figure(num = None, figsize = (6 * nGraphPerRow, 8 * nGraphRow), dpi =
80, facecolor = 'w', edgecolor = 'k')
    for i in range(min(nCol, nGraphShown)):
        plt.subplot(nGraphRow, nGraphPerRow, i + 1)
        columnDf = df.iloc[:, i]
        if (not np.issubdtype(type(columnDf.iloc[0]), np.number)):
            valueCounts = columnDf.value_counts()
            valueCounts.plot.bar()
        else:
            columnDf.hist()
        plt.ylabel('counts')
        plt.xticks(rotation = 90)
        plt.title(f'{columnNames[i]} (column {i})')
    plt.tight_layout(pad = 1.0, w_pad = 1.0, h_pad = 1.0)
    plt.show()
# Scatter and density plots[For visualization of dataset]
def plotScatterMatrix(df1, plotSize, textSize):
    df1 = df1.select_dtypes(include =[np.number]) # keep only numerical columns
    # Remove rows and columns that would lead to df being singular
    df1 = df1.dropna('columns')
    df1 = df1[[col for col in df if df[col].nunique() > 1]] # keep columns where there
are more than 1 unique values
    columnNames = list(df)
    if len(columnNames) > 10: # reduce the number of columns for matrix inversion
of kernel density plots
        columnNames = columnNames[:10]
    df1 = df1[columnNames]
    ax = pd.plotting.scatter_matrix(df1, alpha=0.75, figsize=[plotSize, plotSize],
diagonal='kde')
    corrs = df1.corr().values
    for i, j in zip(*plt.np.triu_indices_from(ax, k = 1)):
        ax[i, j].annotate('Corr. coef = %.3f' % corrs[i, j], (0.8, 0.2), xycoords='axes
fraction', ha='center', va='center', size=textSize)
    plt.suptitle('Scatter and Density Plot')
    plt.show()

plotPerColumnDistribution(df, 10, 5) # Listing 10 rows of bargraph and dividing it
in 5 column
plotScatterMatrix(df, 20, 10) # Listing 20 rows of scattermatrix and dividing it in
10 column
# Creating Test dataset
X= df[l1]
```

```
y = df[["prognosis"]]
np.ravel(y)
print(X)
# Output of dataset
print(y)
# Creating Testing File
#Reading the  testing.csv file
tr=pd.read_csv("testing.csv")

#Using inbuilt function replace in pandas for replacing the values

tr.replace({'prognosis':{'Fungal infection':0,'Allergy':1,'GERD':2,'Chronic cholestasis':3,'Drug Reaction':4,
   'Peptic ulcer diseae':5,'AIDS':6,'Diabetes ':7,'Gastroenteritis':8,'Bronchial Asthma':9,'Hypertension ':10,
   'Migraine':11,'Cervical spondylosis':12,
   'Paralysis (brain hemorrhage)':13,'Jaundice':14,'Malaria':15,'Chicken pox':16,'Dengue':17,'Typhoid':18,'hepatitis A':19,
   'Hepatitis B':20,'Hepatitis C':21,'Hepatitis D':22,'Hepatitis E':23,'Alcoholic hepatitis':24,'Tuberculosis':25,
   'Common Cold':26,'Pneumonia':27,'Dimorphic hemmorhoids(piles)':28,'Heart attack':29,'Varicose veins':30,'Hypothyroidism':31,
   'Hyperthyroidism':32,'Hypoglycemia':33,'Osteoarthristis':34,'Arthritis':35,
   '(vertigo) Paroymsal  Positional Vertigo':36,'Acne':37,'Urinary tract infection':38,'Psoriasis':39,
   'Impetigo':40}},inplace=True)

# Printing head of test file
tr.head()
# Visualize plotPerColumnDistribution
plotPerColumnDistribution(tr, 10, 5)
# Visualize plotScatterMatrix
plotScatterMatrix(tr, 20, 10)
# Creating test dataset
```

```
X_test= tr[l1]
y_test = tr[["prognosis"]]
np.ravel(y_test)
print(X_test)
# Printing Test dataset
print(y_test)
```

To build the precision of the model, we utilized three distinctive algorithms which are as per the following

* Decision Tree algorithm
* Random Forest algorithm
* KNearestNeighbour algorithm
* Naive Bayes algorithm

TO visualize result of different algorithm two functions are used ie scatterplot and scatterinput

```
# For predecting scatterplt function is called
#list1 = DF['prognosis'].unique()
def scatterplt(disea):
    x = ((DF.loc[disea]).sum())#total sum of symptom reported for given disease
    x.drop(x[x==0].index,inplace=True)#droping symptoms with values 0
    print(x.values)
    y = x.keys()#storing nameof symptoms in y
    print(len(x))
    print(len(y))
    plt.title(disea)
    plt.scatter(y,x.values)
    plt.show()



# While taking symptoms scatterinp function is called
def scatterinp(sym1,sym2,sym3,sym4,sym5):
    x = [sym1,sym2,sym3,sym4,sym5]#storing input symptoms in y
    y = [0,0,0,0,0]#creating and giving values to the input symptoms
    if(sym1!='Select Here'):
        y[0]=1
    if(sym2!='Select Here'):
        y[1]=1
    if(sym3!='Select Here'):
        y[2]=1
    if(sym4!='Select Here'):
        y[3]=1
    if(sym5!='Select Here'):
        y[4]=1
    print(x)
    print(y)
    plt.scatter(x,y)
    plt.show()
# Decision Tree Algorithm
root = Tk()
pred1=StringVar()
def DecisionTree():
    if len(NameEn.get()) == 0:
        pred1.set(" ")
        comp=messagebox.askokcancel("System","Kindly Fill the Name")
        if comp:
            root.mainloop()
    elif((Symptom1.get()=="Select Here") or (Symptom2.get()=="Select Here")):
        pred1.set(" ")
        sym=messagebox.askokcancel("System","Kindly Fill atleast first two
Symptoms")
        if sym:
            root.mainloop()
    else:
        from sklearn import tree

        clf3 = tree.DecisionTreeClassifier()
        clf3 = clf3.fit(X,y)
        from sklearn.metrics import classification_report,confusion_matrix,accuracy_score
        y_pred=clf3.predict(X_test)
        print("Decision Tree")
        print("Accuracy")
        print(accuracy_score(y_test, y_pred))
        print(accuracy_score(y_test, y_pred,normalize=False))
        print("Confusion matrix")
        conf_matrix=confusion_matrix(y_test,y_pred)
        print(conf_matrix)

        psymptoms = [Symptom1.get(),Symptom2.get(),Symptom3.get(),Symptom4.get(),Symptom5.get()]

        for k in range(0,len(l1)):
            for z in psymptoms:
                if(z==l1[k]):
                    l2[k]=1

        inputtest = [l2]
        predict = clf3.predict(inputtest)
        predicted=predict[0]

        h='no'
        for a in range(0,len(disease)):
            if(predicted == a):
                h='yes'
                break


        if (h=='yes'):
            pred1.set(" ")
            pred1.set(disease[a])
        else:
            pred1.set(" ")
            pred1.set("Not Found")
        #Creating the database if not exists named as database.db and creating table if not exists named as DecisionTree using sqlite3
        import sqlite3
        conn = sqlite3.connect('database.db')
        c = conn.cursor()
        c.execute("CREATE TABLE IF NOT EXISTS DecisionTree(Name StringVar,Symtom1 StringVar,Symtom2 StringVar,Symtom3 StringVar,Symtom4 TEXT,Symtom5 TEXT,Disease StringVar)")
        c.execute("INSERT INTO DecisionTree(Name,Symtom1,Symtom2,Symtom3,Symtom4,Symtom5,Disease) VALUES(?,?,?,?,?,?,?)",(NameEn.get(),Symptom1.get(),Symptom2.get(),Symptom3.get(),Symptom4.get(),Symptom5.get(),pred1.get()))
        conn.commit()
        c.close()
        conn.close()

        #printing scatter plot of input symptoms
        #printing scatter plot of disease predicted vs its symptoms

scatterinp(Symptom1.get(),Symptom2.get(),Symptom3.get(),Symptom4.get(),Symptom5.get())
        scatterplt(pred1.get())
# Random Forest Algorithm
pred2=StringVar()
def randomforest():
    if len(NameEn.get()) == 0:
        pred1.set(" ")
        comp=messagebox.askokcancel("System","Kindly Fill the Name")
        if comp:
            root.mainloop()
    elif((Symptom1.get()=="Select Here") or (Symptom2.get()=="Select Here")):
        pred1.set(" ")
        sym=messagebox.askokcancel("System","Kindly Fill atleast first two
Symptoms")
        if sym:
            root.mainloop()
    else:
        from sklearn.ensemble import RandomForestClassifier
        clf4 = RandomForestClassifier(n_estimators=100)
        clf4 = clf4.fit(X,np.ravel(y))

        # calculating accuracy
        from sklearn.metrics import
classification_report,confusion_matrix,accuracy_score
        y_pred=clf4.predict(X_test)
        print("Random Forest")
        print("Accuracy")
        print(accuracy_score(y_test, y_pred))
        print(accuracy_score(y_test, y_pred,normalize=False))
        print("Confusion matrix")
        conf_matrix=confusion_matrix(y_test,y_pred)
        print(conf_matrix)

        psymptoms =
[Symptom1.get(),Symptom2.get(),Symptom3.get(),Symptom4.get(),Symptom5.get
()]

        for k in range(0,len(l1)):
            for z in psymptoms:
                if(z==l1[k]):
                    l2[k]=1

        inputtest = [l2]
        predict = clf4.predict(inputtest)
        predicted=predict[0]
        h='no'
        for a in range(0,len(disease)):
            if(predicted == a):
                h='yes'
                break
        if (h=='yes'):
            pred2.set(" ")
            pred2.set(disease[a])
        else:
            pred2.set(" ")
            pred2.set("Not Found")
         #Creating the database if not exists named as database.db and creating table if not exists named as RandomForest using sqlite3
        import sqlite3
        conn = sqlite3.connect('database.db')
        c = conn.cursor()
        c.execute("CREATE TABLE IF NOT EXISTS RandomForest(Name StringVar,Symtom1 StringVar,Symtom2 StringVar,Symtom3 StringVar,Symtom4 TEXT,Symtom5 TEXT,Disease StringVar)")
        c.execute("INSERT INTO RandomForest(Name,Symtom1,Symtom2,Symtom3,Symtom4,Symtom5,Disease) VALUES(?,?,?,?,?,?,?)",(NameEn.get(),Symptom1.get(),Symptom2.get(),Symptom3.get(),Symptom4.get(),Symptom5.get(),pred2.get()))
        conn.commit()
        c.close()
        conn.close()
        #printing scatter plot of disease predicted vs its symptoms
        scatterplt(pred2.get())
# KNearestNeighbour Algorithm
pred4=StringVar()
def KNN():
    if len(NameEn.get()) == 0:
        pred1.set(" ")
        comp=messagebox.askokcancel("System","Kindly Fill the Name")
        if comp:
            root.mainloop()
    elif((Symptom1.get()=="Select Here") or (Symptom2.get()=="Select Here")):
        pred1.set(" ")
        sym=messagebox.askokcancel("System","Kindly Fill atleast first two
Symptoms")
        if sym:
            root.mainloop()
    else:
        from sklearn.neighbors import KNeighborsClassifier
        knn=KNeighborsClassifier(n_neighbors=5,metric='minkowski',p=2)
        knn=knn.fit(X,np.ravel(y))

        from sklearn.metrics import
classification_report,confusion_matrix,accuracy_score
        y_pred=knn.predict(X_test)
        print("kNearest Neighbour")
        print("Accuracy")
        print(accuracy_score(y_test, y_pred))
        print(accuracy_score(y_test, y_pred,normalize=False))
        print("Confusion matrix")
        conf_matrix=confusion_matrix(y_test,y_pred)
        print(conf_matrix)

        psymptoms =
[Symptom1.get(),Symptom2.get(),Symptom3.get(),Symptom4.get(),Symptom5.get
()]

        for k in range(0,len(l1)):
            for z in psymptoms:
                if(z==l1[k]):
                    l2[k]=1

        inputtest = [l2]
        predict = knn.predict(inputtest)
        predicted=predict[0]

        h='no'
        for a in range(0,len(disease)):
            if(predicted == a):
                h='yes'
                break



        if (h=='yes'):
            pred4.set(" ")
            pred4.set(disease[a])
        else:
            pred4.set(" ")
            pred4.set("Not Found")
         #Creating the database if not exists named as database.db and creating table if not exists named as KNearestNeighbour using sqlite3
        import sqlite3
        conn = sqlite3.connect('database.db')
        c = conn.cursor()
        c.execute("CREATE TABLE IF NOT EXISTS KNearestNeighbour(Name StringVar,Symtom1 StringVar,Symtom2 StringVar,Symtom3 StringVar,Symtom4 TEXT,Symtom5 TEXT,Disease StringVar)")
        c.execute("INSERT INTO KNearestNeighbour(Name,Symtom1,Symtom2,Symtom3,Symtom4,Symtom5,Disease) VALUES(?,?,?,?,?,?,?)",(NameEn.get(),Symptom1.get(),Symptom2.get(),Symptom3.get(),Symptom4.get(),Symptom5.get(),pred4.get()))
        conn.commit()
        c.close()
        conn.close()
        #printing scatter plot of disease predicted vs its symptoms

        scatterplt(pred4.get())
```

```
# Naive Bayes Algorithm
pred3=StringVar()
def NaiveBayes():
    if len(NameEn.get()) == 0:
        pred1.set(" ")
        comp=messagebox.askokcancel("System","Kindly Fill the Name")
        if comp:
            root.mainloop()
    elif((Symptom1.get()=="Select Here") or (Symptom2.get()=="Select Here")):
        pred1.set(" ")
        sym=messagebox.askokcancel("System","Kindly Fill atleast first two
Symptoms")
        if sym:
            root.mainloop()
    else:
        from sklearn.naive_bayes import GaussianNB
        gnb = GaussianNB()
        gnb=gnb.fit(X,np.ravel(y))

        from sklearn.metrics import
classification_report,confusion_matrix,accuracy_score
        y_pred=gnb.predict(X_test)
        print("Naive Bayes")
        print("Accuracy")
        print(accuracy_score(y_test, y_pred))
        print(accuracy_score(y_test, y_pred,normalize=False))
        print("Confusion matrix")
        conf_matrix=confusion_matrix(y_test,y_pred)
        print(conf_matrix)

        psymptoms =
[Symptom1.get(),Symptom2.get(),Symptom3.get(),Symptom4.get(),Symptom5.get
()]
        for k in range(0,len(l1)):
            for z in psymptoms:
                if(z==l1[k]):
                    l2[k]=1

        inputtest = [l2]
        predict = gnb.predict(inputtest)
        predicted=predict[0]

        h='no'
        for a in range(0,len(disease)):
            if(predicted == a):
                h='yes'
                break
        if (h=='yes'):
            pred3.set(" ")
            pred3.set(disease[a])
        else:
            pred3.set(" ")
            pred3.set("Not Found")
         #Creating the database if not exists named as database.db and creating table if not exists named as NaiveBayes using sqlite3
        import sqlite3
        conn = sqlite3.connect('database.db')
        c = conn.cursor()
        c.execute("CREATE TABLE IF NOT EXISTS NaiveBayes(Name StringVar,Symtom1 StringVar,Symtom2 StringVar,Symtom3 StringVar,Symtom4 TEXT,Symtom5 TEXT,Disease StringVar)")
        c.execute("INSERT INTO NaiveBayes(Name,Symtom1,Symtom2,Symtom3,Symtom4,Symtom5,Disease) VALUES(?,?,?,?,?,?,?)",(NameEn.get(),Symptom1.get(),Symptom2.get(),Symptom3.get(),Symptom4.get(),Symptom5.get(),pred3.get()))
        conn.commit()
        c.close()
        conn.close()
        #printing scatter plot of disease predicted vs its symptoms
        scatterplt(pred3.get())
# Building Graphical User Interface Tkinter
# Tk class is used to create a root window
root.configure(background='#B0C4DE')
root.title('Smart Disease Predictor System')
root.resizable(0,0)
# Functions for Main_menu

def help():
    showinfo("Contact me", "upendra.gupta.7543@gmail.com")

def about():
    showinfo("Smart Disease Predictor System", "Developed by Cureya Team \n
Upendra Gupta(Team Leader)\n Aneesha Sengupta \n Prateek Singh \n Vasvi
Agarwal \n Earavelly Sriharshitha \n Madhuri Kumari \n Nikita Verma")

def quitApp():
    qExit=messagebox.askyesno("System","Do you want to exit the system")

    if qExit:
        root.destroy()
        exit()
# Icons
root.wm_iconbitmap("cureya.ico")
# Menubar
MenuBar = Menu(root)
# Main_menu
Main_menu = Menu(MenuBar, tearoff=0)
Main_menu.add_command(label="Help", command=help)
Main_menu.add_command(label="About Developer", command=about)
Main_menu.add_command(label = "Exit", command=quitApp)
MenuBar.add_cascade(label="Menu", menu=Main_menu)
# Exit Menu
```

```
root.config(menu=MenuBar)
# Taking Inputs
Symptom1 = StringVar()
Symptom1.set("Select Here")

Symptom2 = StringVar()
Symptom2.set("Select Here")

Symptom3 = StringVar()
Symptom3.set("Select Here")

Symptom4 = StringVar()
Symptom4.set("Select Here")

Symptom5 = StringVar()
Symptom5.set("Select Here")
Name = StringVar()
# Function to Reset the given inputs to initial position
prev_win=None
def Reset():
    global prev_win

    Symptom1.set("Select Here")
    Symptom2.set("Select Here")
    Symptom3.set("Select Here")
    Symptom4.set("Select Here")
    Symptom5.set("Select Here")
    NameEn.delete(first=0,last=100)
    pred1.set(" ")
    pred2.set(" ")
    pred3.set(" ")
    pred4.set(" ")
    try:
        prev_win.destroy()
        prev_win=None
    except AttributeError:
        pass
# Headings for the GUI written at the top of GUI
w2 = Label(root, justify=LEFT, text="Smart Disease Predictor System",
fg="White", bg="#B0C4DE")
w2.config(font=("Times",30,"bold italic"))
w2.grid(row=1, column=0, columnspan=2, padx=100)
# Label for the name
NameLb = Label(root, text="Name of the Patient *", fg="Black", bg="#B0C4DE")
NameLb.config(font=("Times",15,"bold italic"))
NameLb.grid(row=6, column=0, pady=15, sticky=W)
# Creating Labels for the symtoms

S1Lb = Label(root, text="Symptom 1 *", fg="Black", bg="#B0C4DE")
S1Lb.config(font=("Times",15,"bold italic"))
S1Lb.grid(row=7, column=0, pady=10, sticky=W)

S2Lb = Label(root, text="Symptom 2 *", fg="Black", bg="#B0C4DE")
S2Lb.config(font=("Times",15,"bold italic"))
S2Lb.grid(row=8, column=0, pady=10, sticky=W)

S3Lb = Label(root, text="Symptom 3", fg="Black", bg="#B0C4DE")
S3Lb.config(font=("Times",15,"bold italic"))
S3Lb.grid(row=9, column=0, pady=10, sticky=W)

S4Lb = Label(root, text="Symptom 4", fg="Black", bg="#B0C4DE")
S4Lb.config(font=("Times",15,"bold italic"))
S4Lb.grid(row=10, column=0, pady=10, sticky=W)

S5Lb = Label(root, text="Symptom 5", fg="Black", bg="#B0C4DE")
S5Lb.config(font=("Times",15,"bold italic"))
S5Lb.grid(row=11, column=0, pady=10, sticky=W)
```

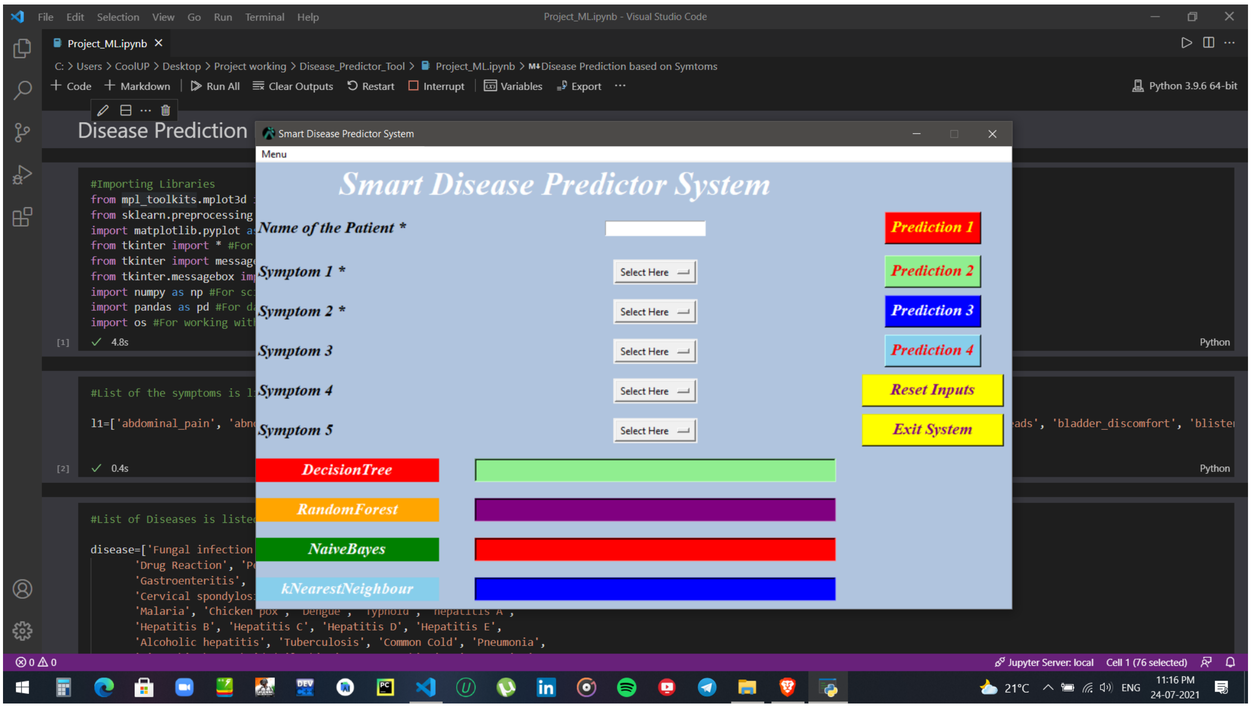
```
# Labels for the different algorithms
lrLb = Label(root, text="DecisionTree", fg="white", bg="red", width = 20)
lrLb.config(font=("Times",15,"bold italic"))
lrLb.grid(row=15, column=0, pady=10,sticky=W)

destreeLb = Label(root, text="RandomForest", fg="White", bg="Orange", width = 20)
destreeLb.config(font=("Times",15,"bold italic"))
destreeLb.grid(row=17, column=0, pady=10, sticky=W)

ranfLb = Label(root, text="NaiveBayes", fg="White", bg="green", width = 20)
ranfLb.config(font=("Times",15,"bold italic"))
ranfLb.grid(row=19, column=0, pady=10, sticky=W)

knnLb = Label(root, text="kNearestNeighbour", fg="White", bg="Sky Blue", width = 20)
knnLb.config(font=("Times",15,"bold italic"))
knnLb.grid(row=21, column=0, pady=10, sticky=W)
OPTIONS = sorted(l1)
# Taking name as input from user
NameEn = Entry(root, textvariable=Name)
NameEn.grid(row=6, column=1)
# Taking Symptoms as input from the dropdown from the user
S1 = OptionMenu(root, Symptom1,*OPTIONS)
S1.grid(row=7, column=1)

S2 = OptionMenu(root, Symptom2,*OPTIONS)
S2.grid(row=8, column=1)

S3 = OptionMenu(root, Symptom3,*OPTIONS)
S3.grid(row=9, column=1)

S4 = OptionMenu(root, Symptom4,*OPTIONS)
S4.grid(row=10, column=1)
```

```
S5 = OptionMenu(root, Symptom5,*OPTIONS)
S5.grid(row=11, column=1)
# Buttons for predicting the disease using different algorithms
dst = Button(root, text="Prediction 1", command=DecisionTree,bg="Red",fg="yellow")
dst.config(font=("Times",15,"bold italic"))
dst.grid(row=6, column=3,padx=10)

rnf = Button(root, text="Prediction 2", command=randomforest,bg="Light green",fg="red")
rnf.config(font=("Times",15,"bold italic"))
rnf.grid(row=7, column=3,padx=10)

lr = Button(root, text="Prediction 3", command=NaiveBayes,bg="Blue",fg="white")
lr.config(font=("Times",15,"bold italic"))
lr.grid(row=8, column=3,padx=10)

kn = Button(root, text="Prediction 4", command=KNN,bg="sky blue",fg="red")
kn.config(font=("Times",15,"bold italic"))
kn.grid(row=9, column=3,padx=10)

rs = Button(root,text="Reset Inputs", command=Reset,bg="yellow",fg="purple",width=15)
rs.config(font=("Times",15,"bold italic"))
rs.grid(row=10,column=3,padx=10)

ex = Button(root,text="Exit System", command=quitApp,bg="yellow",fg="purple",width=15)
ex.config(font=("Times",15,"bold italic"))
ex.grid(row=11,column=3,padx=10)

# Showing the output of different algorithms
```

```
t1=Label(root,font=("Times",15,"bold italic"),text="Decision
Tree",height=1,bg="Light green"
        ,width=40,fg="red",textvariable=pred1,relief="sunken").grid(row=15,
column=1, padx=10)

t2=Label(root,font=("Times",15,"bold italic"),text="Random
Forest",height=1,bg="Purple"
        ,width=40,fg="white",textvariable=pred2,relief="sunken").grid(row=17,
column=1, padx=10)

t3=Label(root,font=("Times",15,"bold italic"),text="Naive
Bayes",height=1,bg="red"
        ,width=40,fg="orange",textvariable=pred3,relief="sunken").grid(row=19,
column=1, padx=10)

t4=Label(root,font=("Times",15,"bold italic"),text="kNearest
Neighbour",height=1,bg="Blue"
        ,width=40,fg="yellow",textvariable=pred4,relief="sunken").grid(row=21,
column=1, padx=10)
# Calling this function because the application is ready to run
root.mainloop()
```

9. Result:

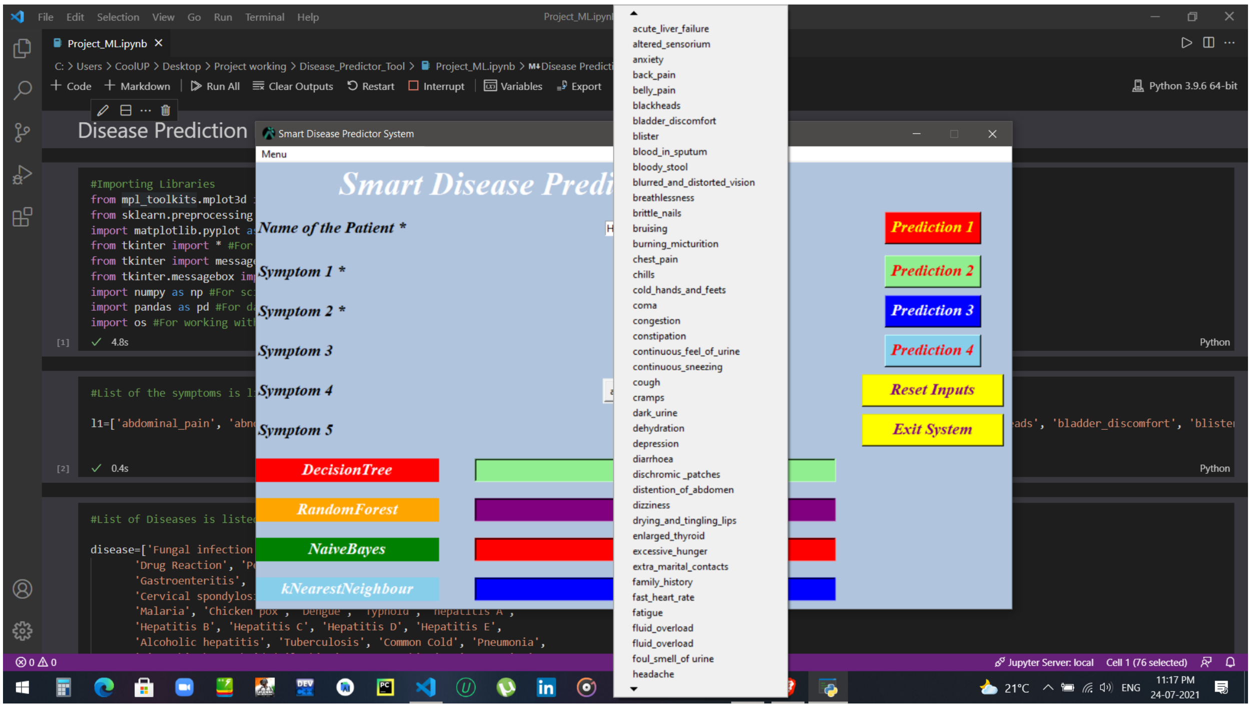
1. Snapshot of Result:

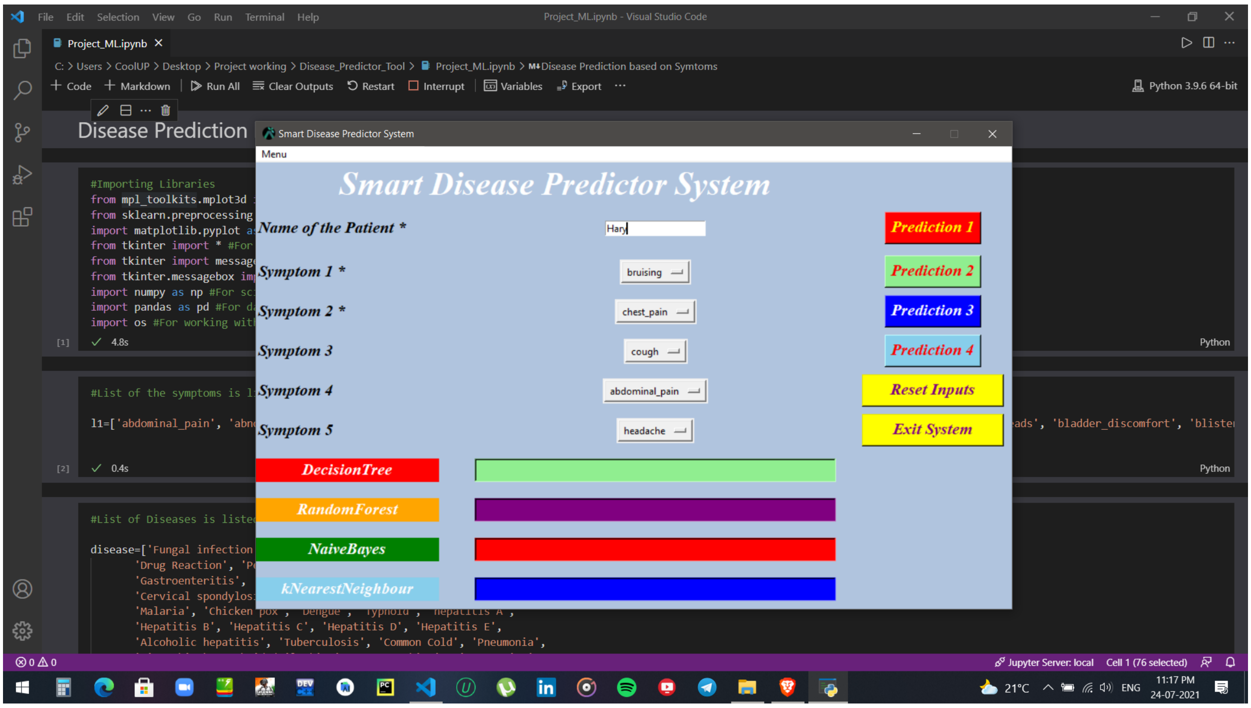


```
#Importing Libraries
from mpl_toolkits.mplot3d
from sklearn.preprocessing
import matplotlib.pyplot a
from tkinter import * #For
from tkinter import message
from tkinter.messagebox im
import numpy as np #For sc
import pandas as pd #For d
import os #For working wit
```

```
#List of the symptoms is l

l1=['abdominal_pain', 'abn                ads', 'bladder_discomfort', 'bliste
```

```
#List of Diseases is liste

disease=['Fungal infection
        'Drug Reaction', 'P
        'Gastroenteritis',
        'Cervical spondylos
        'Malaria', 'Chicken pox', 'Dengue', 'Typhoid', 'hepatitis A',
        'Hepatitis B', 'Hepatitis C', 'Hepatitis D', 'Hepatitis E',
        'Alcoholic hepatitis', 'Tuberculosis', 'Common Cold', 'Pneumonia',
```




File Edit Selection View Go Run Terminal Help
Project_ML.ipynb - Visual Studio Code
Project_ML.ipynb
C: > Users > CoolUP > Desktop > Project working > Disease_Predictor_Tool > Project_ML.ipynb > M↓Disease Prediction based on Symtoms
+ Code + Markdown Run All Clear Outputs Restart Interrupt Variables Export
Python 3.9.6 64-bit
Disease Prediction
Smart Disease Predictor System
Menu
Prediction 1
Prediction 2
Prediction 3
Prediction 4
Reset Inputs
Exit System
DecisionTree
RandomForest
NaiveBayes
kNearestNeighbour
Jupyter Server: local Cell 1 (76 selected)
21°C ENG 11:17 PM 24-07-2021

Smart Disease Predictor System

Name of the Patient * Hary

Symptom 1 * bruising

Symptom 2 * chest_pain

Symptom 3 cough

Symptom 4 abdominal_pain

Symptom 5 headache

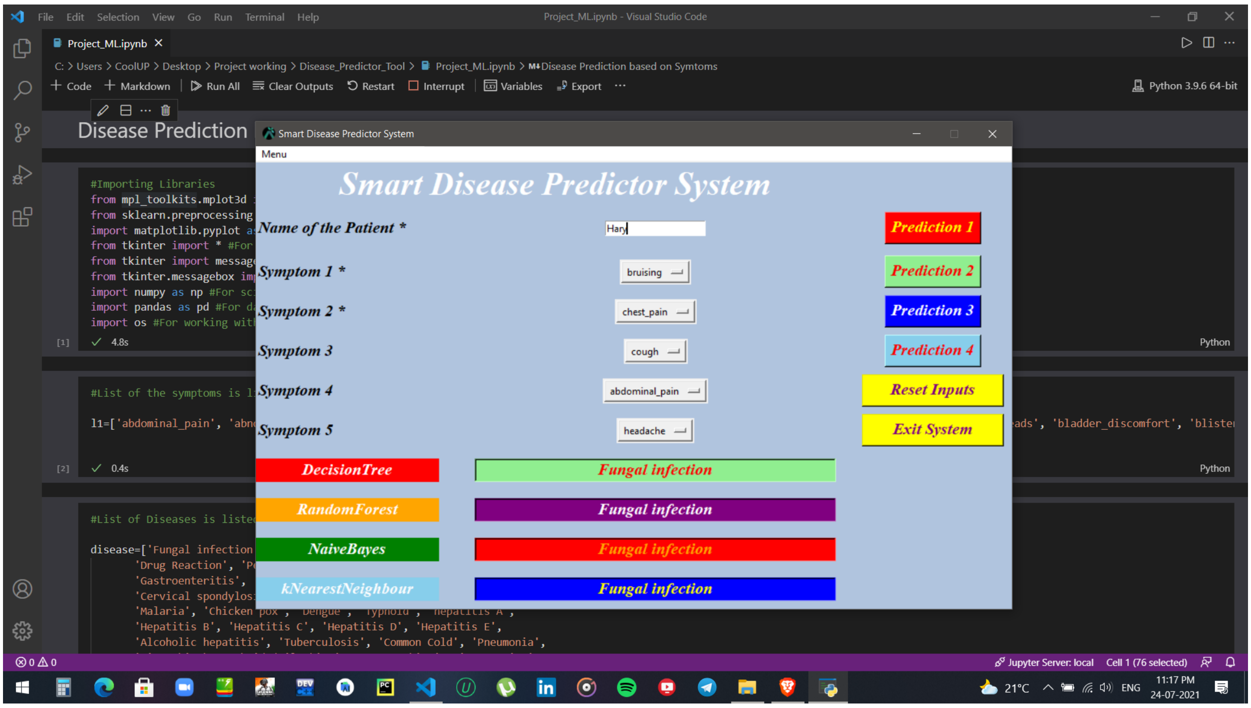
```
#Importing Libraries
from mpl_toolkits.mplot3d
from sklearn.preprocessing
import matplotlib.pyplot a
from tkinter import * #For
from tkinter import messag
from tkinter.messagebox im
import numpy as np #For sc
import pandas as pd #For d
import os #For working wit
```

[1] ✓ 4.8s Python

```
#List of the symptoms is l

l1=['abdominal_pain', 'abn                                       ads', 'bladder_discomfort', 'bliste
```

[2] ✓ 0.4s Python

```
#List of Diseases is liste

disease=['Fungal infection
        'Drug Reaction', 'P
        'Gastroenteritis',
        'Cervical spondylos
        'Malaria', 'Chicken pox', 'Dengue', 'Typhoid', 'hepatitis A',
        'Hepatitis B', 'Hepatitis C', 'Hepatitis D', 'Hepatitis E',
        'Alcoholic hepatitis', 'Tuberculosis', 'Common Cold', 'Pneumonia',
```


File Edit Selection View Go Run Terminal Help
Project_ML.ipynb - Visual Studio Code
Project_ML.ipynb
C: > Users > CoolUP > Desktop > Project working > Disease_Predictor_Tool > Project_ML.ipynb > M↓Disease Prediction based on Symtoms
+ Code + Markdown Run All Clear Outputs Restart Interrupt Variables Export
Python 3.9.6 64-bit
Disease Prediction
Smart Disease Predictor System
Menu
Prediction 1
Prediction 2
Prediction 3
Prediction 4
Reset Inputs
Exit System
Jupyter Server: local Cell 1 (76 selected)
21°C ENG 11:17 PM 24-07-2021

Smart Disease Predictor System

Name of the Patient * Hary

Symptom 1 * bruising

Symptom 2 * chest_pain

Symptom 3 cough

Symptom 4 abdominal_pain

Symptom 5 headache

DecisionTree Fungal infection

RandomForest Fungal infection

NaiveBayes Fungal infection

kNearestNeighbour Fungal infection

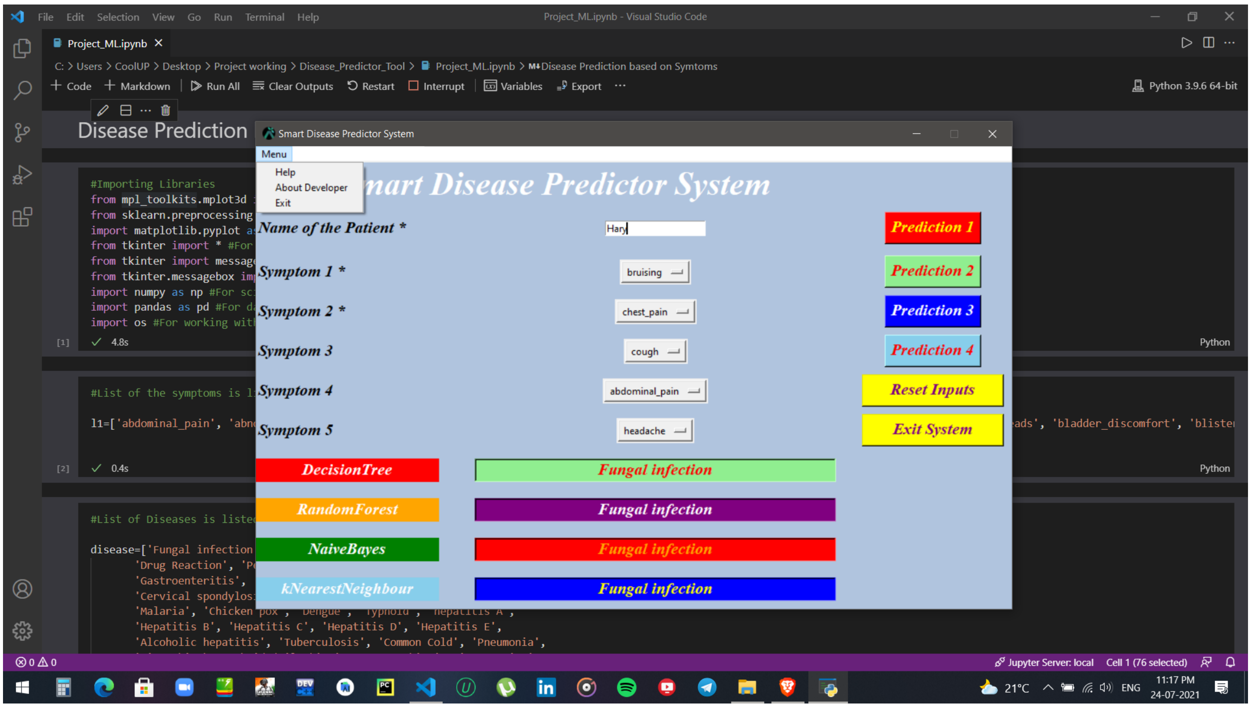
```
#Importing Libraries
from mpl_toolkits.mplot3d
from sklearn.preprocessing
import matplotlib.pyplot a
from tkinter import * #For
from tkinter import messag
from tkinter.messagebox im
import numpy as np #For sc
import pandas as pd #For d
import os #For working wit
```

[1] ✓ 4.8s Python

```
#List of the symptoms is l

l1=['abdominal_pain', 'abn                                       ads', 'bladder_discomfort', 'bliste
```

[2] ✓ 0.4s Python

```
#List of Diseases is liste

disease=['Fungal infection
        'Drug Reaction', 'P
        'Gastroenteritis',
        'Cervical spondylos
        'Malaria', 'Chicken pox', 'Dengue', 'Typhoid', 'hepatitis A',
        'Hepatitis B', 'Hepatitis C', 'Hepatitis D', 'Hepatitis E',
        'Alcoholic hepatitis', 'Tuberculosis', 'Common Cold', 'Pneumonia',
```


File Edit Selection View Go Run Terminal Help
Project_ML.ipynb - Visual Studio Code
C: > Users > CoolUP > Desktop > Project working > Disease_Predictor_Tool > Project_ML.ipynb > M↓Disease Prediction based on Symtoms
+ Code + Markdown | Run All Clear Outputs Restart Interrupt Variables Export
Python 3.9.6 64-bit
Smart Disease Predictor System
Menu
Help
About Developer
Exit
Prediction 1
Prediction 2
Prediction 3
Prediction 4
Reset Inputs
Exit System
DecisionTree
RandomForest
NaiveBayes
kNearestNeighbour
Jupyter Server: local Cell 1 (76 selected)
11:17 PM 24-07-2021

Disease Prediction

```
#Importing Libraries
from mpl_toolkits.mplot3d
from sklearn.preprocessing
import matplotlib.pyplot a
from tkinter import * #For
from tkinter import messag
from tkinter.messagebox im
import numpy as np #For sc
import pandas as pd #For d
import os #For working wit
```

[1] ✓ 4.8s

```
#List of the symptoms is l

l1=['abdominal_pain', 'abn                ads', 'bladder_discomfort', 'bliste
```

[2] ✓ 0.4s

```
#List of Diseases is liste

disease=['Fungal infection
        'Drug Reaction', 'P
        'Gastroenteritis',
        'Cervical spondylos
        'Malaria', 'Chicken pox', 'Dengue', 'Typhoid', 'hepatitis A',
        'Hepatitis B', 'Hepatitis C', 'Hepatitis D', 'Hepatitis E',
        'Alcoholic hepatitis', 'Tuberculosis', 'Common Cold', 'Pneumonia',
```

Smart Disease Predictor System

Name of the Patient * Hary

Symptom 1 * bruising

Symptom 2 * chest_pain

Symptom 3 cough

Symptom 4 abdominal_pain

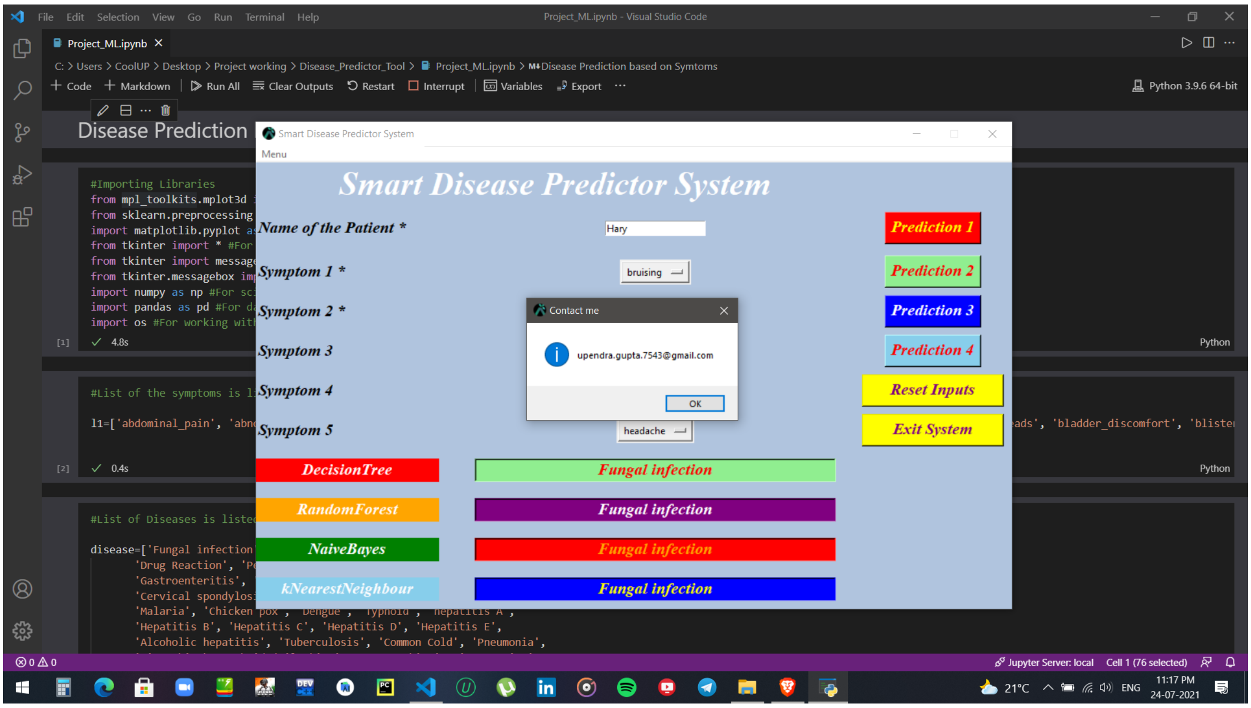
Symptom 5 headache

DecisionTree: Fungal infection

RandomForest: Fungal infection

NaiveBayes: Fungal infection

kNearestNeighbour: Fungal infection


File Edit Selection View Go Run Terminal Help
Project_ML.ipynb - Visual Studio Code
C: > Users > CoolUP > Desktop > Project working > Disease_Predictor_Tool > Project_ML.ipynb > M↓Disease Prediction based on Symtoms
+ Code + Markdown | Run All Clear Outputs Restart Interrupt Variables Export
Python 3.9.6 64-bit
Smart Disease Predictor System
Menu
Prediction 1
Prediction 2
Prediction 3
Prediction 4
Reset Inputs
Exit System
DecisionTree
RandomForest
NaiveBayes
kNearestNeighbour
OK
Jupyter Server: local Cell 1 (76 selected)
11:17 PM 24-07-2021

Smart Disease Predictor System

Name of the Patient * Hary

Symptom 1 * bruising

Symptom 2 *

Symptom 3

Symptom 4

Symptom 5 headache

Contact me

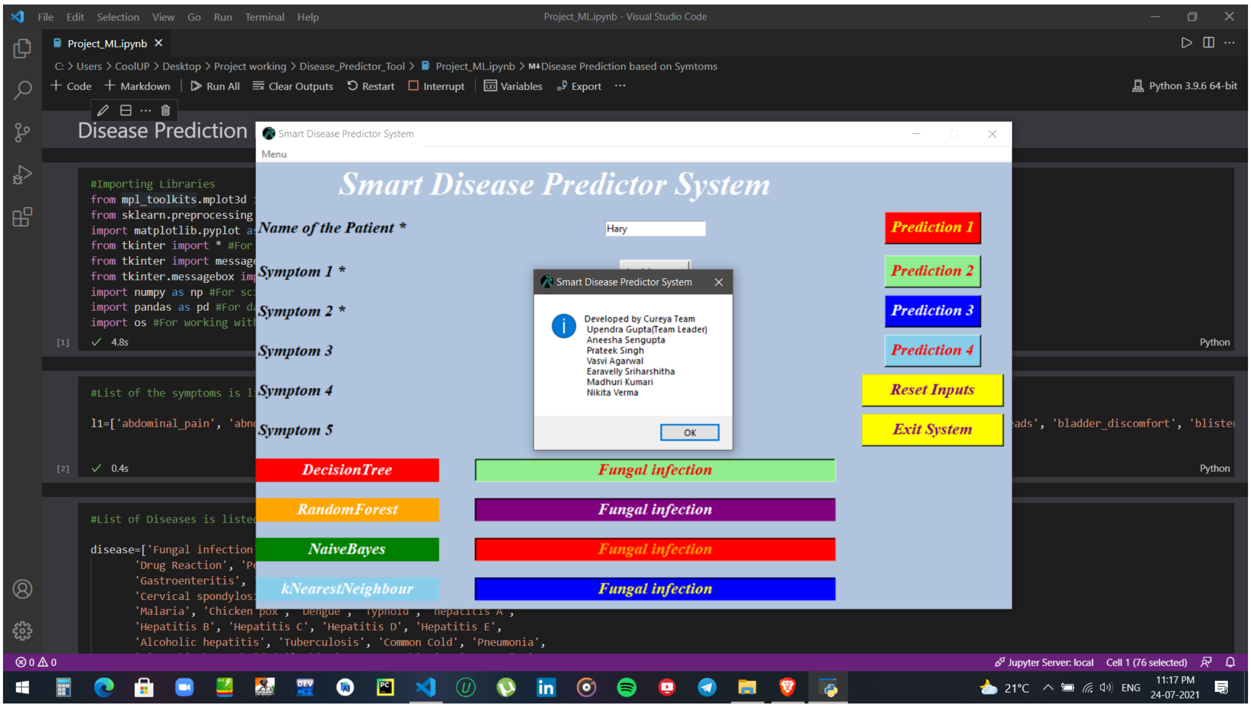
upendra.gupta.7543@gmail.com

DecisionTree: Fungal infection

RandomForest: Fungal infection

NaiveBayes: Fungal infection

kNearestNeighbour: Fungal infection


File Edit Selection View Go Run Terminal Help
Project_ML.ipynb - Visual Studio Code
Project_ML.ipynb
C: > Users > CoolUP > Desktop > Project working > Disease_Predictor_Tool > Project_ML.ipynb > M↓Disease Prediction based on Symtoms
+ Code + Markdown Run All Clear Outputs Restart Interrupt Variables Export
Python 3.9.6 64-bit
Disease Prediction
Menu
Jupyter Server: local Cell 1 (76 selected)
21°C ENG 11:17 PM 24-07-2021

```
#Importing Libraries
from mpl_toolkits.mplot3d i
from sklearn.preprocessing
import matplotlib.pyplot as
from tkinter import * #For
from tkinter import message
from tkinter.messagebox imp
import numpy as np #For sci
import pandas as pd #For da
import os #For working with
```

[1] ✓ 4.8s

```
#List of the symptoms is li

l1=['abdominal_pain', 'abno                    ads', 'bladder_discomfort', 'blister
```

[2] ✓ 0.4s

```
#List of Diseases is listed

disease=['Fungal infection'
        'Drug Reaction', 'Pe
        'Gastroenteritis',
        'Cervical spondylosi
        'Malaria', 'Chicken pox',
        'Hepatitis B', 'Hepatitis C', 'Hepatitis D', 'Hepatitis E',
        'Alcoholic hepatitis', 'Tuberculosis', 'Common Cold', 'Pneumonia',
```

Smart Disease Predictor System

Smart Disease Predictor System

Name of the Patient * Hary

Symptom 1 *

Symptom 2 *

Symptom 3

Symptom 4

Symptom 5

Prediction 1

Prediction 2

Prediction 3

Prediction 4

Reset Inputs

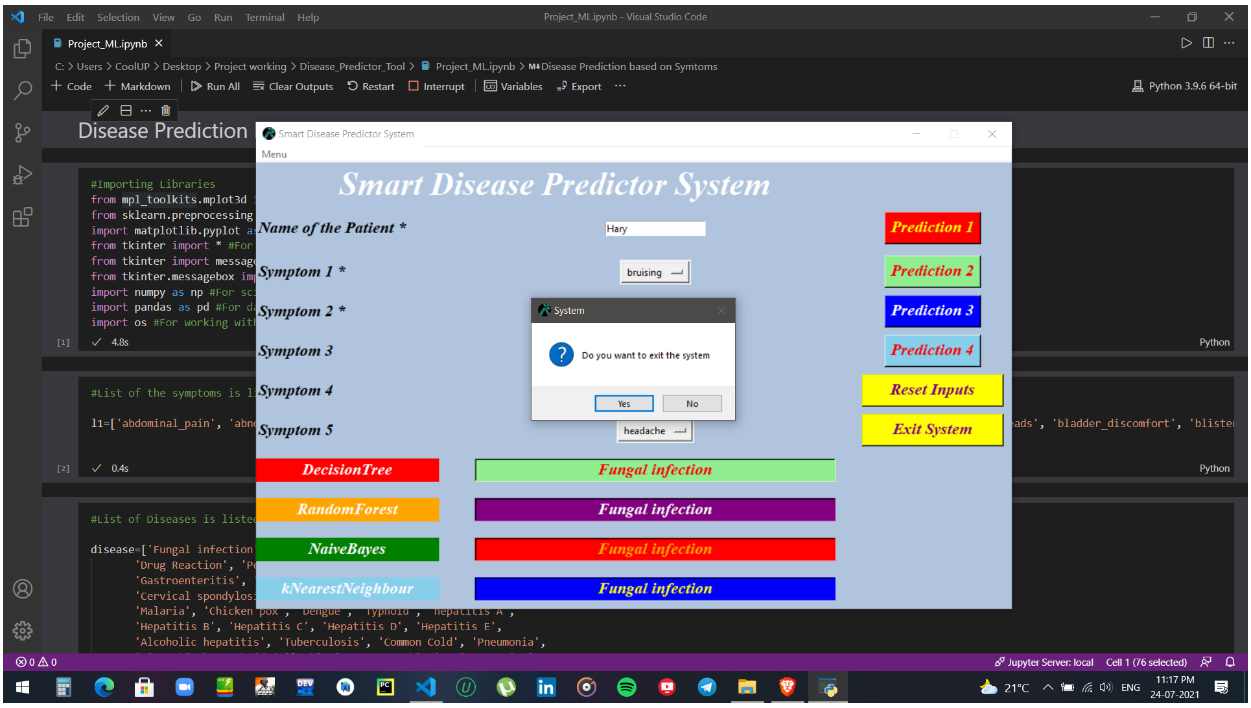
Exit System

DecisionTree	Fungal infection
RandomForest	Fungal infection
NaiveBayes	Fungal infection
kNearestNeighbour	Fungal infection

Smart Disease Predictor System

Developed by Cureya Team
Upendra Gupta(Team Leader)
Aneesha Sengupta
Prateek Singh
Vasvi Agarwal
Earavelly Sriharshitha
Madhuri Kumari
Nikita Verma

OK


File Edit Selection View Go Run Terminal Help
Project_ML.ipynb - Visual Studio Code
Project_ML.ipynb
C: > Users > CoolUP > Desktop > Project working > Disease_Predictor_Tool > Project_ML.ipynb > M↓Disease Prediction based on Symtoms
+ Code + Markdown Run All Clear Outputs Restart Interrupt Variables Export
Python 3.9.6 64-bit
Disease Prediction
Menu
Jupyter Server: local Cell 1 (76 selected)
21°C ENG 11:17 PM 24-07-2021

Smart Disease Predictor System

Smart Disease Predictor System

Name of the Patient * Hary

Symptom 1 * bruising

Symptom 2 *

Symptom 3

Symptom 4

Symptom 5 headache

Prediction 1

Prediction 2

Prediction 3

Prediction 4

Reset Inputs

Exit System

DecisionTree	Fungal infection
RandomForest	Fungal infection
NaiveBayes	Fungal infection
kNearestNeighbour	Fungal infection

System

Do you want to exit the system

Yes No

6. Advantage and Disadvantages of Model:

1. Advantages:

1. The advantage of this system is that the initial consultation cost of doctor fees can be avoided.
2. various data mining techniques are clearly explained.
3. It can produce fast analysis reports, operational efficiency and reduce operational cost.
4. It helps physicians to identify best treatments for particular diseases.
5. There are various algorithms and techniques such as Classification, Clustering, Regression, Artificial intelligence, neural networks, Association rules and Decision trees. The advantage is that various data mining techniques are clearly explained.
6. The goal of this concept is targeted the simplest data is stored into the space of medical massive data
7. The medical data is securely stored and used in many places.

2. Disadvantages:

- the efficiency in detecting the symptoms or symptom mapping.
- maximum accuracy is not achieved in prediction.
- There are also some of the disadvantages such as data ownership problems, privacy and security related issues for human data administration etc.
- This feature is only applicable for the structured data so it is not good in disease description.
- It predicts only the patient related information.

- It selects the best feasible, but not previously checks the possibility.
- The risk prediction is depends on the different feature of medical data

7. Conclusion & Future Scope:

7.1 Conclusion:

Health care plays a major role with the benefit of information technology.The proposed automatic disease prediction system has explored knowledge using big data techniques and provides great potential to overcome those issues and improves clinical prediction almost perfectly. Along with the symptoms by applying a naive bayes algorithm, the patient‟s history from EHR has been evaluated to make our results better. Although this model could accurately predict some diseases, it is still hard for some other diseases, especially those that face complexity in diagnosis. The diagnosis of the disease is easy for doctors and medication can be provided on time. The stages of various diseases can be calculated accurately and according to the patients can be treated. The system reduces the human effort, cost and time constraint and increases the diagnostic accuracy. The overall mission of system development is to make the primary treatment quickly and easily with the use of technology. As it is said prevention is better than cure so the system will help the patient to let them know what they are suffering from till the doctor reacts to it. In future paramedical recommendation and online consultation with the doctors can be made for more convenience.

The proposed study offers health experts a more productive and advantageous route for patients to make forecasts on particular databases we got from outer territory. The objective for the paper is to learn and examine the upgraded strategies of putting away and handling tremendous arrangement of data in health sector

7.2 Future Scope:

Here the scope of the project is that integration of clinical decision support with computer-based patient records could reduce medical errors, enhance patient safety, decrease unwanted practice variation, and improve patient outcomes. This suggestion is promising as data modeling and analysis tools, e.gData mining has the potential to generate a knowledge-rich environment which can help to significantly improve the quality of clinical decisions.

In the future, by the advancement in the field of IT sector, data mining will be much more advanced and can mine different knowledge hidden in medical data. Due to the limit of time and lack of domain knowledge in healthcare insurance, we cannot implement all the ideas in our minds. We will definitely continue our research and design more AI tools that help improve the efficiency of the medical system and balance the medical resources throughout the whole country.

8. References:

1. F. Jiang, Y. Jiang, H. Zhi et al., “Artificial intelligence in healthcare: past, present and future,” *Stroke and Vascular Neurology*, vol. 2, no. 4, pp. 230–243, 2017.
2. View at: Publisher Site | Google Scholar
3. Sidey-Gibbons, A. M. Jenni, and C. J. Sidey-Gibbons, “Machine learning in medicine: a practical introduction,” *BMC Medical Research Methodology*, vol. 19, 2019.
4. View at: Publisher Site | Google Scholar
5. T. Davenport and R. Kalakota, “The potential for artificial intelligence in healthcare,” *Future Healthcare Journal*, vol. 6, no. 2, pp. 94–98, 2019.
6. View at: Publisher Site | Google Scholar
7. J. Keto, H. Ventola, J. Jokelainen et al., “Cardiovascular disease risk factors in relation to smoking behaviour and history: a population-based cohort study,” *Open Heart*, vol. 3, no. 2, 2016.
8. View at: Publisher Site | Google Scholar
9. A. B. Olokoba, O. A. Obateru, and L. B. Olokoba, “Type 2 diabetes mellitus: a review of current trends,” *Oman Medical Journal*, vol. 27, no. 4, pp. 269–273, 2012.
10. View at: Publisher Site | Google Scholar
11. D. Singh and V. Kumar, “Single image defogging by gain gradient image filter,” *Science China Information Sciences*, vol. 62, no. 7, pp. 1–3, 2019.
12. View at: Publisher Site | Google Scholar
13. I. Sartzetakis, K. Christodoulopoulos, and E. Varvarigos, “Accurate quality of transmission estimation with machine learning,” *Journal of Optical Communications and Networking*, vol. 11, no. 3, pp. 140–150, 2019.
14. View at: Publisher Site | Google Scholar
15. S. Otoum, B. Kantarci, and H. T. Mouftah, “On the feasibility of deep learning in sensor network intrusion detection,” *IEEE Networking Letters*, vol. 1, no. 2, pp. 68–71, 2019.
16. View at: Publisher Site | Google Scholar
17. M. Z. Ali, M. N. S. K. Shabbir, X. Liang, Y. Zhang, and T. Hu, “Machine learning-based fault diagnosis for single- and multi-faults in induction motors using measured stator currents and vibration signals,” *IEEE Transactions on Industry Applications*, vol. 55, no. 3, pp. 2378–2391, 2019.

18. View at: Publisher Site | Google Scholar
19. P. Mishra, V. Varadharajan, U. Tupakula, and E. S. Pilli, “A detailed investigation and analysis of using machine learning techniques for intrusion detection,” *IEEE Communications Surveys & Tutorials*, vol. 21, no. 1, pp. 686–728, 2019.
20. View at: Publisher Site | Google Scholar
21. J. Song, F. Dong, J. Zhao, H. Wang, Z. He, and L. Wang, “An efficient multiobjective design optimization method for a PMSM based on an extreme learning machine,” *IEEE Transactions on Industrial Electronics*, vol. 66, no. 2, pp. 1001–1011, 2019.
22. View at: Publisher Site | Google Scholar
23. R. Razavi-Far, E. Hallaji, M. Farajzadeh-Zanjani et al., “Information fusion and semi-supervised deep learning scheme for diagnosing gear faults in induction machine systems,” *IEEE Transactions on Industrial Electronics*, vol. 66, no. 8, pp. 6331–6342, 2019.
24. View at: Publisher Site | Google Scholar
25. S. Ma, J. Dai, S. Lu et al., “Signal demodulation with machine learning methods for physical layer visible light communications: prototype platform, open dataset, and algorithms,” *IEEE Access*, vol. 7, pp. 30588–30598, 2019.
26. View at: Publisher Site | Google Scholar
27. Y. Shen, Y. Shi, J. Zhang, and K. B. Letaief, “LORM: learning to optimize for resource management in wireless networks with few training samples,” *IEEE Transactions on Wireless Communications*, vol. 19, no. 1, pp. 665–679, 2020.
28. View at: Publisher Site | Google Scholar
29. S. Wang, T. Tuor, T. Salonidis et al., “Adaptive federated learning in resource constrained edge computing systems,” *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1205–1221, 2019.
30. View at: Publisher Site | Google Scholar
31. M. Chen, U. Challita, W. Saad, C. Yin, and M. Debbah, “Artificial neural networks-based machine learning for wireless networks: a tutorial,” *IEEE Communications Surveys & Tutorials*, vol. 21, no. 4, pp. 3039–3071, 2019.
32. View at: Publisher Site | Google Scholar
33. S. J. Nawaz, S. K. Sharma, S. Wyne, M. N. Patwary, and M. Asaduzzaman, “Quantum machine learning for 6g communication networks: state-of-the-art and vision for the future,” *IEEE Access*, vol. 7, pp. 46317–46350, 2019.
34. View at: Publisher Site | Google Scholar

35. T. F. Lima, H. Peng, A. N. Tait et al., “Machine learning with neuromorphic photonics,” *Journal of Lightwave Technology*, vol. 37, pp. 1515–1534, 2019.
36. View at: Google Scholar
37. A. Chelli and M. Pätzold, “A machine learning approach for fall detection and daily living activity recognition,” *IEEE Access*, vol. 7, pp. 38670–38687, 2019.
38. View at: Publisher Site | Google Scholar
39. J. Zhang, Y. Wang, P. Molino, L. Li, and D. S. Ebert, “Manifold: a model-agnostic framework for interpretation and diagnosis of machine learning models,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 25, no. 1, pp. 364–373, 2019.
40. View at: Publisher Site | Google Scholar

Team Members

Name: Upendra Gupta (Team Leader)

Branch : Computer Science

College : Visvesvaraya Technological University

Semester: 6th

Name: Aneesha Sengupta

Branch : Mechanical Engineering

College : Sardar Vallabhbhai National Institute of Technology

Semester: 5th

Name: Prateek Singh

Branch : Information Technology

College : KIET Group of Institutions

Semester: 6th Semester

Name: Vasvi Agarwal

Branch : Mechanical

College : University of Wisconsin-Madison

Semester: 4th Sem

Name: Earavelly Sriharshitha

Branch : Computer Science and Engineering

College : IIT Delhi

Semester: 2nd semester

Name: Madhuri

Branch : Computer Science and Engineering

College : Guru Gobind Singh educational society’s technical campus

Semester: 6 th

Name: Nikita Verma

Branch : Computer Science and Engineering

College : Inderprastha Engineering College

Semester: 6 th