**MULTIPLE DISEASE DETECTION (DIABETES, CHRONIC KIDNEY, LEVER DISEASE, BREAST ,SKIN CANCER)**

|  |  |
| --- | --- |
| **CONTENT** | **PAGE NO** |
| **Abstract** |  |
| **1.INTRODUCTION** |  |
| 1.1 Motivation |  |
| 1.2 Problem Statement |  |
| 1.3 Objective of the Project |  |
| 1.4 Scope |  |
| 1.5 Project Introduction |  |
| **2.LTERATURE SURVEY** |  |
| 2.1 Related Work |  |
| **3. SYSTEM ANALYSIS** |  |
| 3.1 Existing System |  |
| 3.2 Disadvantages |  |
| 3.3 Proposed System |  |
| 3.4 Advantages |  |
| 3.5 work Flow of Proposed system |  |
| **4. REQUIREMENT ANALYSIS** |  |
| 4.1Function and non-functional requirements |  |
| 4.2 Hardware Requirements |  |
| 4.3 Software Requirements |  |
| 4.4 Architecture |  |
| **5. SYSTEM DESIGN** |  |
| 5.1 Introduction of Input design |  |
| 5.2 UML Diagram(class, use case, sequence, collaborative, deployment, activity, ER diagram and Component diagram) |  |
| 5.3 Data Flow Diagram |  |
| **6. IMPLEMENTATION AND RESULTS** |  |
| 6.1 Modules |  |
| 6.2 Output Screens |  |
| 1. **PYTHON ENVIRONMENT** |  |
| 7.1 Software Installation |  |
| 7.2 About Python Language |  |
| 1. **SYSTEM STUDY AND TESTING** |  |
| 8.1 Feasibility study |  |
| 8.2 Types of test & Test Cases |  |
| **9. CONCLUSION** |  |
| **10. FUTURE ENHANCEMENT** |  |
| **11. REFERENCES** |  |

**OBJECTIVE**

The main objective of this research is to classify the diseases using machine learning by entering features regarding disease, it will predict the patient has disease or not.

**MOTIVATION:**

The motivation behind "Multiple Disease Detection (Diabetes, Chronic Kidney, Liver Disease, Breast, Skin Cancer)" is the urgent need for a comprehensive and accessible healthcare solution. As the global burden of these diseases continues to rise, early detection and intervention become paramount. Leveraging advanced technologies like machine learning and medical data analysis, this project aims to empower individuals and healthcare providers with a versatile tool for timely disease identification. By combining multiple disease detection capabilities, we aspire to enhance public health outcomes, reduce healthcare costs, and ultimately improve the quality of life for millions of individuals by enabling proactive and personalized healthcare management.

**PROBLEM STATEMENT**

The problem statement is to develop a machine learning model that can accurately detect multiple diseases including diabetes, chronic kidney disease, liver disease, and breast and skin cancer. The model should be trained on relevant medical data such as patient demographics, medical history, symptoms, and diagnostic test results. The goal is to create a system that can aid in early disease detection and diagnosis, potentially improving patient outcomes and reducing healthcare costs. The model should also be robust and generalizable, able to perform accurately on new patient data from different populations and geographic regions.

**ABSTRACT**

Globally, there is a substantial unmet need to diagnose various diseases effectively. The complexity of the different disease mechanisms and underlying symptoms of the patient population presents massive challenges to developing the early diagnosis tool and effective treatment. Machine Learning (ML) an area of Artificial Intelligence (AI), enables researchers, physicians, and patients to solve some of these issues. Based on relevant research, this review explains how Machine Learning (ML). Early detection and diagnosis of diseases such as diabetes, chronic kidney disease, liver disease, and breast cancer is crucial for improving patient outcomes and reducing healthcare costs. In recent years, machine learning has emerged as a promising tool for disease detection and diagnosis. In this project, we aim to develop a machine learning model for multiple disease detection, which can aid in early disease diagnosis and treatment. The proposed model will be trained on a large dataset of medical records, which includes patient demographics, medical history, symptoms, and diagnostic test results. The dataset will be carefully curated and pre-processed to ensure high data quality and completeness. We will also incorporate relevant features such as genetic markers and lifestyle factors, which have been shown to influence disease risk.

**Keywords**: Machine Learning, Decision tree, Adaboost , Xgboost and Catboost , CNN and ML techniques, evaluation.

**Objective of the Project:**

The objective of this project is to develop a unified, AI-driven healthcare system for the early detection of multiple critical diseases, including Diabetes, Chronic Kidney Disease, Liver Disease, Breast Cancer, and Skin Cancer. Utilizing advanced machine learning algorithms and medical data analysis, our goal is to create a user-friendly platform that can assist both individuals and healthcare providers in timely disease identification. This system aims to improve healthcare outcomes by enabling proactive interventions, reducing treatment costs, and enhancing the overall well-being and longevity of patients.

**Scope:**

The scope of this project is extensive, encompassing the development of a versatile healthcare system capable of detecting multiple diseases, including Diabetes, Chronic Kidney Disease, Liver Disease, Breast Cancer, and Skin Cancer. This system will leverage cutting-edge technologies such as machine learning, data analytics, and medical imaging analysis. It will be designed to facilitate early disease identification, risk assessment, and personalized healthcare recommendations. The project will involve the integration of diverse medical data sources and the creation of user-friendly interfaces for patients and healthcare professionals. The ultimate scope is to enhance public health, improve disease management, and contribute to early intervention and better healthcare outcomes for individuals globally.

**INTRODUCTION\**

Multiple Disease Detection is an emerging technology that aims to detect the presence of multiple diseases simultaneously. The technology utilizes advanced data analytics and machine learning algorithms to analyze data from multiple sources, such as medical records, lab reports, and imaging studies, to provide a comprehensive diagnosis of multiple diseases.The technology has shown great promise in the early detection and management of several chronic diseases, including Diabetes, Chronic Kidney Disease, Liver Disease, and Breast Cancer. These diseases are among the leading causes of morbidity and mortality worldwide, and early detection and intervention can significantly improve patient outcomes.Diabetes is a chronic disease characterized by high levels of blood sugar. Early detection of diabetes can prevent complications such as blindness, kidney disease, and nerve damage. Chronic Kidney Disease is a progressive disease that can lead to kidney failure if left untreated. Early detection and management can slow down the progression of the disease and prevent kidney failure.Liver Disease is a broad term used to describe any condition that affects the liver. Early detection of liver disease can prevent liver damage and improve liver function. Breast Cancer is the most common cancer in women worldwide. Early detection and treatment can significantly improve the chances of survival.The multiple disease detection technology utilizes advanced algorithms to analyze patient data and provide a comprehensive diagnosis of multiple diseases. This can significantly improve the accuracy and speed of diagnosis, enabling healthcare providers to provide timely and effective treatment.In conclusion, Multiple Disease Detection technology has great potential in the early detection and management of chronic diseases such as Diabetes, Chronic Kidney Disease, Liver Disease, and Breast, Skin Cancer. Early detection and treatment of these diseases can significantly improve patient outcomes and reduce healthcare costs. As technology continues to advance, we can expect to see even more sophisticated tools and techniques for detecting and managing multiple diseases. Within the aging population, the frequency of cancer is increasing dramatically. In addition, multiple genetic and environmental factors lead to common multifactorial diseases, including cardiovascular disease, chronic kidney disease, chronic obstructive pulmonary disease, and metabolic-associated fatty liver disease. In recent years, there has been a growing awareness of the connection between cancer and multifactorial diseases, as well as how one can affect the other, resulting in a vicious cycle. Although the exact mechanistic explanations behind this remain to be fully explored, some progress has been made in uncovering the common pathologic mechanisms. In this review, we focus on the nature of the link between cancer and common multifactorial conditions, as well as specific shared mechanisms, some of which may represent either preventive or therapeutic targets. Rather than organ-specific interactions, we herein focus on the shared mechanisms among the multifactorial diseases, which may explain the increased cancer risk. More research on this subject will highlight the significance of developing new drugs that target multiple systems rather than just one disease.

**LITERATURE SURVEY**

**[1]. J. L. Scully, ‘‘What is a disease?’’ EMBO Rep., vol. 5, no. 7, pp. 650–653,**

**2004.**

Chronic diseases are increasing in prevalence and mortality worldwide. Early diagnosis has therefore become an important research area to enhance patient survival rates. Several research studies have reported classification approaches for specific disease prediction. In this paper, we propose a novel augmented artificial intelligence approach using an artificial neural network (ANN) with particle swarm optimization (PSO) to predict five prevalent chronic diseases including breast cancer, diabetes, heart attack, hepatitis, and kidney disease. Seven classification algorithms are compared to evaluate the proposed model's prediction performance. The ANN prediction model constructed with a PSO based feature extraction approach outperforms other state-of-the-art classification approaches when evaluated with accuracy. Our proposed approach gave the highest accuracy of 99.67%, with the PSO. However, the classification model's performance is found to depend on the attributes of data used for classification. Our results are compared with various chronic disease datasets and shown to outperform other benchmark approaches. In addition, our optimized ANN processing is shown to require less time compared to random forest (RF), deep learning and support vector machine (SVM) based methods. Our study could play a role for early diagnosis of chronic diseases in hospitals, including through development of online diagnosis systems.

**[2]** **R. Leaman, R. Islamaj Dogan, and Z. Lu, ‘‘DNorm: Disease name normalization with pairwise learning to rank,’’ Bioinformatics, vol. 29, no. 22, pp. 2909–2917, Nov. 2013.**

Despite the central role of diseases in biomedical research, there have been much fewer attempts to automatically determine which diseases are mentioned in a text—the task of disease name normalization (DNorm)—compared with other normalization tasks in biomedical text mining research. In this article we introduce the first machine learning approach for DNorm, using the NCBI disease corpus and the MEDIC vocabulary, which combines MeSH® and OMIM. Our method is a high-performing and mathematically principled framework for learning similarities between mentions and concept names directly from training data. The technique is based on pairwise learning to rank, which has not previously been applied to the normalization task but has proven successful in large optimization problems for information retrieval.

**[3] N. Armstrong and P. Hilton, ‘‘Doing diagnosis: Whether and how clinicians use a diagnostic tool of uncertain clinical utility,’’ Social Sci. Med., vol. 120, pp. 208–214, Nov. 2014.**

Diagnosis is fundamental to the practice of medicine and mastery of it is central to the process of both becoming and practicing as a doctor. We focus on diagnosis as a process, in particular from the perspective of clinicians performing it. We explore how UK clinicians exercise discretion about whether and how to use a diagnostic tool (invasive [urodynamic](https://www.sciencedirect.com/topics/medicine-and-dentistry/urodynamics) tests – IUT) for which there is, currently, no clear, high-quality evidence. Interviews were conducted with a purposive sample of 18 clinicians who had previously completed a survey on their use of IUT. Analysis was based on the constant comparative method. Participants tended to be polarised in their view of IUT. While many regarded it as a valuable diagnostic tool that they used frequently and thought was important, others reported using it only infrequently, and some were sceptical of its value in the diagnostic process even if they commonly used it. In addition to the anticipated clinical functions (e.g. adding to understanding of the condition, helping determine best treatment) there were additional, more social, functions that IUT could serve, including fitting in with local practice and helping to defend against possible future litigation. We discern two distinct approaches to the practice of diagnosis: one approach means ‘leaving no stone unturned’ and seeking all available evidence, proven or otherwise; while a second means using clinical judgement to say ‘enough is enough’ and thereby avoid exposing patients to possibly unnecessary tests and potentially wasting scarce healthcare resources.

**[4] R. H. Scheuermann, W. Ceusters, and B. Smith, ‘‘Toward an ontological treatment of disease and diagnosis,’’ Summit Transl. Bioinformat., vol. 2009, p. 116, Mar. 2009**

Many existing biomedical vocabulary standards rest on incomplete, inconsistent or confused accounts of basic terms pertaining to diseases, diagnoses, and clinical phenotypes. Here we outline what we believe to be a logically and biologically coherent framework for the representation of such entities and of the relations between them. We defend a view of disease as involving in every case some physical basis within the organism that bears a disposition toward the execution of pathological processes. We present our view in the form of a list of terms and definitions designed to provide a consistent starting point for the representation of both disease and diagnosis in information systems in the future.

**SYSTEM ANALYSIS & FEASIBILITY STUDY**

**EXISTING SYSTEM**

The existing system of multiple disease detection using machine learning employs algorithms that can analyze medical data and identify patterns to predict the risk of developing diabetes, chronic kidney disease, liver disease, and breast cancer. This system uses various features such as patient demographics, medical history, symptoms, and laboratory test results to generate accurate predictions. By leveraging machine learning techniques, this system can continuously learn and improve its predictive accuracy, providing early detection and potentially life-saving interventions for patients at risk of developing these diseases.

**ADVANTAGES :**

**1. Data Imbalance**: One significant challenge is dealing with imbalanced datasets, particularly for rare diseases like certain types of cancer. Imbalanced data can lead to biased model training and result in lower accuracy for disease detection, especially when the minority class is of interest. Addressing this issue requires careful data preprocessing and potentially the use of advanced techniques like resampling or synthetic data generation.

**2. Interpretability and Trust:** Some machine learning models, especially complex ones like deep neural networks, can be challenging to interpret. Understanding why a model made a specific prediction is crucial, especially in medical contexts. The lack of interpretability can hinder healthcare professionals' trust in the system and may limit its adoption in clinical practice. Developing methods for model explainability and transparency is vital to address this limitation.

**PROPOSED SYSTEM**

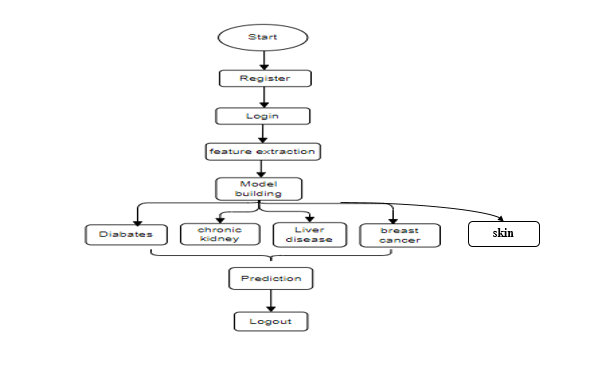
The proposed system of multiple disease detection using machine learning involves the development of a model that can accurately detect the presence of five diseases - diabetes, chronic kidney disease, liver disease, and breast , skin cancer - using patient data. The system will use various machine learning algorithms to analyze patient data such as medical history, symptoms, and laboratory test results to identify patterns and predict the likelihood of disease. The system will provide clinicians with accurate and timely information to improve patient outcomes, reduce costs, and optimize healthcare delivery. This will be accomplished through a user-friendly interface and integration with electronic medical records.

**Advantages**:

**1. Early Detection and Timely Intervention:** The system's ability to accurately detect multiple diseases at an early stage can lead to timely interventions and treatments. Early detection often results in better patient outcomes, reduced healthcare costs, and improved quality of life for individuals with these diseases.

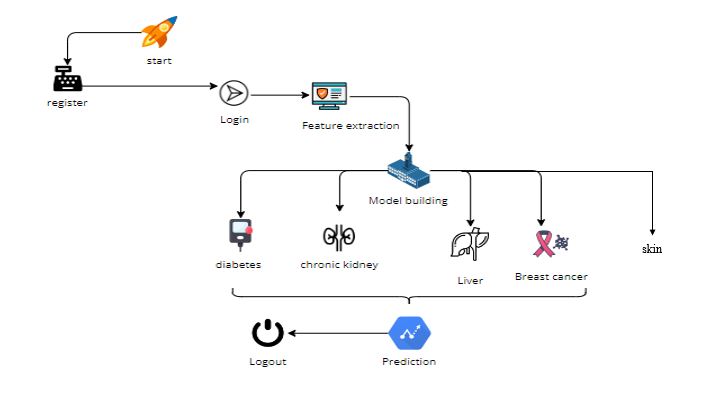
**2. Optimized Healthcare Delivery**: By providing clinicians with accurate and timely information, the system can optimize healthcare delivery. It enables healthcare professionals to prioritize patients at higher risk, allocate resources efficiently, and tailor treatment plans based on individual patient profiles. This can lead to more effective and personalized healthcare services.

**BLOCK DIAGRAM**



**Fig**: Block Diagram

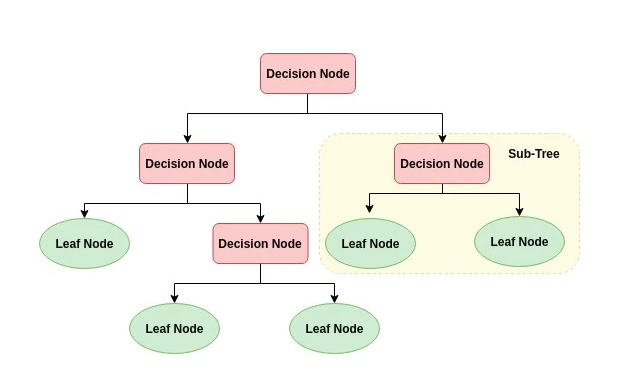
**Architecture:**

****

**METHODOLOGY AND ALGORITHMS:**

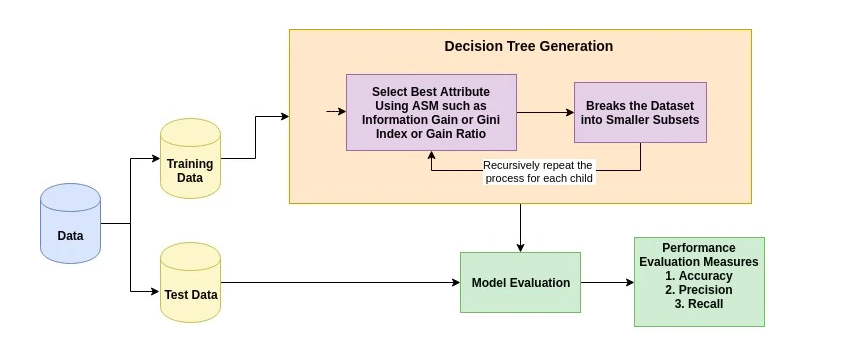
**1. DECISION TREE:**

Decision tree is a flowchart-like tree structure where an internal node represents feature(or attribute), the branch represents a decision rule, and each leaf node represents the outcome. The topmost node in a decision tree is known as the root node. It learns to partition on the basis of the attribute value. It partitions the tree in recursively manner call recursive partitioning. This flowchart-like structure helps you in decision making. It's visualization like a flowchart diagram which easily mimics the human level thinking. That is why decision trees are easy to understand and interpret.

The basic idea behind any decision tree algorithm is as follows:

1. Select the best attribute using Attribute Selection Measures (ASM) to split the records.
2. Make that attribute a decision node and breaks the dataset into smaller subsets.
3. Starts tree building by repeating this process recursively for each child until one of the conditions will match:

* All the tuples belong to the same attribute value.
* There are no more remaining attributes.
* There are no more instances.



**ADABOOST**

AdaBoost algorithm, short for Adaptive Boosting, is a Boosting technique used as an Ensemble Method in Machine Learning. It is called Adaptive Boosting as the weights are re-assigned to each instance, with higher weights assigned to incorrectly classified instances. Boosting is used to reduce bias as well as variance for supervised learning. It works on the principle of learners growing sequentially. Except for the first, each subsequent learner is grown from previously grown learners. In simple words, weak learners are converted into strong ones. The AdaBoost algorithm works on the same principle as boosting with a slight difference. Let’s discuss this difference in detail.First, let us discuss how boosting works. It makes ‘n’ number of decision trees during the data training period. As the first decision tree/model is made, the incorrectly classified record in the first model is given priority. Only these records are sent as input for the second model. The process goes on until we specify a number of base learners we want to create. Remember, repetition of records is allowed with all boosting techniques.This figure shows how the first model is made and errors from the first model are noted by the algorithm. The record which is incorrectly classified is used as input for the next model. This process is repeated until the specified condition is met. As you can see in the figure, there are ‘n’ number of models made by taking the errors from the previous model. This is how boosting works. The models 1,2, 3,…, N are individual models that can be known as decision trees. All types of boosting models work on the same principle. Since we now know the boosting principle, it will be easy to understand the AdaBoost algorithm. Let’s dive into AdaBoost’s working. When the random forest is used, the algorithm makes an ‘n’ number of trees. It makes proper trees that consist of a start node with several leaf nodes. Some trees might be bigger than others, but there is no fixed depth in a random forest. With AdaBoost, however, the algorithm only makes a node with two leaves, known as Stump.The figure here represents the stump. It can be seen clearly that it has only one node with two leaves. These stumps are weak learners and boosting techniques prefer this. The order of stumps is very important in AdaBoost. The error of the first stump influences how other stumps are made. Let’s understand this with an example.

Here’s a sample dataset consisting of only three features where the output is in categorical form. The image shows the actual representation of the dataset. As the output is in binary/categorical form, it becomes a classification problem. In real life, the dataset can have any number of records and features in it. Let us consider 5 datasets for explanation purposes. The output is in categorical form, here in the form of Yes or No. All these records will be assigned a sample weight. The formula used for this is ‘W=1/N’ where N is the number of records. In this dataset, there are only 5 records, so the sample weight becomes 1/5 initially. Every record gets the same weight. In this case, it’s 1/5.

**XGBoost**

XGBoost stands for “Extreme Gradient Boosting”. XGBoost is an optimized distributed gradient boosting library designed to be highly efficient, flexible and portable. It implements Machine Learning algorithms under the Gradient Boosting framework. It provides a parallel tree boosting to solve many data science problems in a fast and accurate way.

**Boosting**

Boosting is an ensemble learning technique to build a strong classifier from several weak classifiers in series. Boosting algorithms play a crucial role in dealing with [bias-variance trade-off](https://www.mygreatlearning.com/blog/bias-variance-trade-off-in-machine-learning/). Unlike bagging algorithms, which only controls for high variance in a model, boosting controls both the aspects (bias & variance) and is considered to be more effective.

Below are the few types of boosting algorithms:

* AdaBoost (Adaptive Boosting)
* [Gradient Boosting](https://www.mygreatlearning.com/blog/gradient-descent/)
* XGBoost
* CatBoost
* Light GBM

**XGBoost**

XGBoost stands for eXtreme Gradient Boosting. It became popular in the recent days and is dominating applied machine learning and Kaggle competitions for structured data because of its scalability.

XGBoost is an extension to gradient boosted decision trees (GBM) and specially designed to improve speed and performance.

**XGBoost Features**

* **Regularized Learning:** Regularization term helps to smooth the final learnt weights to avoid over-fitting. The regularized objective will tend to select a model employing simple and predictive functions.
* **Gradient Tree Boosting**: The tree ensemble model cannot be optimized using traditional optimization methods in Euclidean space. Instead, the model is trained in an additive manner.
* **Shrinkage and Column Subsampling:** Besides the regularized objective, two additional techniques are used to further prevent over fitting. The first technique is shrinkage introduced by Friedman. Shrinkage scales newly added weights by a factor η after each step of tree boosting. Similar to a learning rate in stochastic optimization, shrinkage reduces the influence of each tree and leaves space for future trees to improve the model.

The second technique is the column (feature) subsampling. This technique is used in Random Forest. Column sub-sampling prevents over-fitting even more so than the traditional row sub-sampling. The usage of column sub-samples also speeds up computations of the parallel algorithm.

**SPLITTING ALGORITHMS**

* **Exact Greedy Algorithm:** The main problem in tree learning is to find the best split. This algorithm enumerates over all the possible splits on all the features. It is computationally demanding to enumerate all the possible splits for continuous features.
* **Approximate Algorithm:**The exact greedy algorithm is very powerful since it enumerates over all possible splitting points greedily. However, it is impossible to efficiently do so when the data does not fit entirely into memory. Approximate Algorithm proposes candidate splitting points according to percentiles of feature distribution. The algorithm then maps the continuous features into buckets split by these candidate points, aggregates the statistics and finds the best solution among proposals based on the aggregated statistics.
* **Weighted Quantile Sketch:** One important step in the approximate algorithm is to propose candidate split points. XGBoost has a distributed weighted quantile sketch algorithm to effectively handle weighted data.
* **Sparsity-aware Split Finding:** In many real-world problems, it is quite common for the input x to be sparse. There are multiple possible causes for sparsity:

1. Presence of missing values in the data
2. Frequent zero entries in the statistics
3. Artifacts of feature engineering such as one-hot encoding

It is important to make the algorithm aware of the sparsity pattern in the data. XGBoost handles all sparsity patterns in a unified way.

**System Features**

1. Parallelization of tree construction using all of your CPU cores during training. Collecting statistics for each column can be parallelized, giving us a parallel algorithm for split finding.
2. Cache-aware Access: XGBoost has been designed to make optimal use of hardware. This is done by allocating internal buffers in each thread, where the gradient statistics can be stored.
3. Blocks for Out-of-core Computation for very large datasets that don’t fit into memory.
4. Distributed Computing for training very large models using a cluster of machines.
5. Column Block for Parallel Learning. The most time-consuming part of tree learning is to get the data into sorted order. In order to reduce the cost of sorting, the data is stored in the column blocks in sorted order in compressed format.

**Goals of XGBoost**

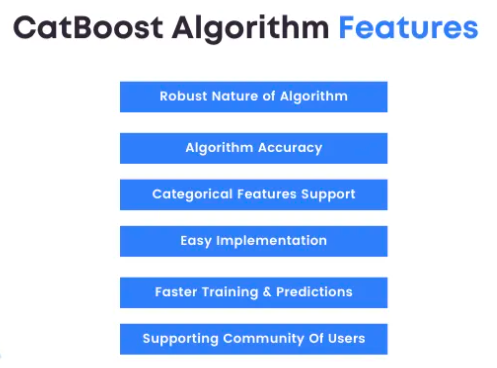
1. Execution Speed: XGBoost was almost always faster than the other benchmarked implementations from R, Python Spark and H2O and it is really faster when compared to the other algorithms.
2. Model Performance: XGBoost dominates structured or tabular datasets on classification and regression predictive modelling problems.

**Conclusion**

XGBoost is a faster algorithm when compared to other algorithms because of its parallel and distributed computing. XGBoost is developed with both deep considerations in terms of systems optimization and principles in machine learning. The goal of this library is to push the extreme of the computation limits of machines to provide a scalable, portable and accurate library.

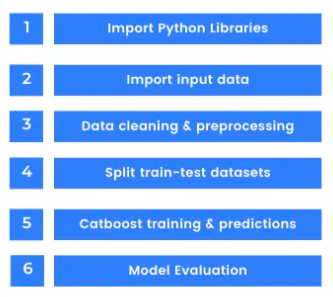
**Cat Boost:**

CatBoost is a high-performance open source library for gradient boosting on decision trees. CatBoost is an algorithm for gradient boosting on decision trees. It is developed by Yandex researchers and engineers, and is used for search, recommendation systems, personal assistant, self-driving cars, weather prediction and many other tasks at Yandex and in other companies, including CERN, Cloudflare and Careem taxi. It is in open-source and can be used by anyone. Catboost, the new kid on the block, has been around for a little more than a year now, and it is already threatening XGBoost, LightGBM.



Catboost achieves the best results on the benchmark, and that’s great.  
Though, when you look at datasets where categorical features play a large role, this improvement becomes significant and undeniable.

**Implementation:**



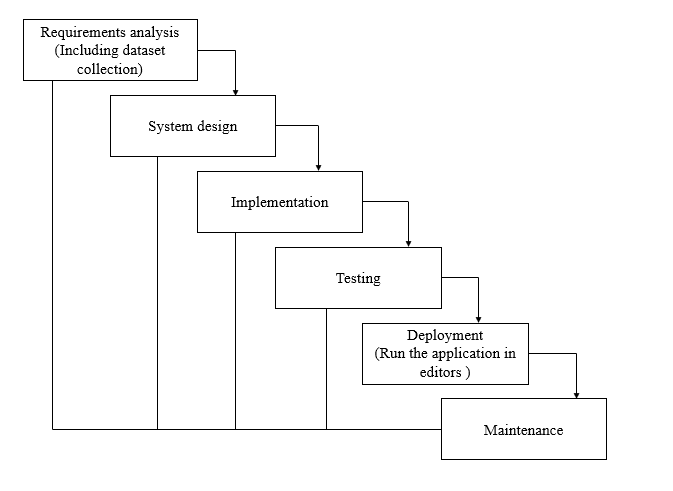
While training time can take up longer than other GBDT implementations, prediction time is 13–16 times faster than the other libraries according to the Yandex benchmark. Catboost’s default parameters are a better starting point than in other GBDT algorithms and it  is good news for beginners who want a plug and play model to start experience tree ensembles or Kaggle competitions. Some more noteworthy advancements by Catboost are the features interactions, object importance and the snapshot support. In addition to classification and regression, Catboost supports ranking out of the box.

**CNN**:

Convolutional Neural Networks (CNNs) are a class of deep learning models designed for processing structured grid-like data, such as images and videos. They excel at feature extraction through a series of convolutional layers, where small filters slide over input data to detect patterns and spatial relationships. CNNs are equipped with pooling layers to reduce dimensionality and fully connected layers for classification or regression tasks. Their hierarchical architecture allows them to automatically learn increasingly complex features, making them particularly suited for tasks like image recognition, object detection, and image generation. CNNs have revolutionized computer vision and have applications in diverse domains beyond image analysis, including natural language processing and drug discovery.

**SOFTWARE DEVELOPMENT LIFE CYCLE – SDLC:**

In our project we use waterfall model as our software development cycle because of its step-by-step procedure while implementing.



**Fig1**: Waterfall Model

* **Requirement Gathering and analysis** − All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.
* **System Design** − the requirement specifications from first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.
* **Implementation** − with inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.
* **Integration and Testing** − All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.
* **Deployment of system** − Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.
* **Maintenance** − There are some issues which come up in the client environment. To fix those issues, patches are released. Also, to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

**FEASIBILITY STUDY**

The feasibility of the project is analysed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

* ECONOMICAL FEASIBILITY
* TECHNICAL FEASIBILITY
* SOCIAL FEASIBILITY

**Economic feasibility:**

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus, the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

### Technical feasibility:

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

**Social feasibility:**

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

**SYSTEM REQUIREMENTS SPECIFICATION**

**Functional and non-functional requirements:**

Requirement’s analysis is very critical process that enables the success of a system or software project to be assessed. Requirements are generally split into two types: Functional and non-functional requirements.

**Functional Requirements**: These are the requirements that the end user specifically demands as basic facilities that the system should offer. All these functionalities need to be necessarily incorporated into the system as a part of the contract. These are represented or stated in the form of input to be given to the system, the operation performed and the output expected. They are basically the requirements stated by the user which one can see directly in the final product, unlike the non-functional requirements.

Examples of functional requirements:

1. Authentication of user whenever he/she logs into the system
2. System shutdown in case of a cyber-attack
3. A verification email is sent to user whenever he/she register for the first time on some software system.

**Non-functional requirements**: These are basically the quality constraints that the system must satisfy according to the project contract. The priority or extent to which these factors are implemented varies from one project to other. They are also called non-behavioral requirements.  
They basically deal with issues like:

* Portability
* Security
* Maintainability
* Reliability
* Scalability
* Performance
* Reusability
* Flexibility

Examples of non-functional requirements:

1. Emails should be sent with a latency of no greater than 12 hours from such an activity.
2. The processing of each request should be done within 10 seconds
3. The site should load in 3 seconds whenever of simultaneous users are > 10000

**SOFTWARE AND HARDWARE REQUIREMENTS**

**H/W Configuration:**

Operating system : Windows 7 or 7+

RAM : 8 GB

Hard disc or SSD : More than 500 GB

Processor : Intel 3rd generation or high or Ryzen with 8 GB Ram

**S/W Configuration:**

Software’s : Python 3.6 or high version

IDE : PyCharm.

Framework : Flask, pandas, numpy and Scikit-Learn

**Learning Outcome:**

1. Regarding Machine Learning
2. Supervised Machine Learning
3. Classification Technique
4. About Pycharm
5. Flask Frame work

**SYSTEM DESIGN:**

## **Input Design:**

In an information system, input is the raw data that is processed to produce output. During the input design, the developers must consider the input devices such as PC, MICR, OMR, etc.

Therefore, the quality of system input determines the quality of system output. Well-designed input forms and screens have following properties −

* It should serve specific purpose effectively such as storing, recording, and retrieving the information.
* It ensures proper completion with accuracy.
* It should be easy to fill and straightforward.
* It should focus on user’s attention, consistency, and simplicity.
* All these objectives are obtained using the knowledge of basic design principles regarding −
  + What are the inputs needed for the system?
  + How end users respond to different elements of forms and screens.

### Objectives for Input Design:

The objectives of input design are −

* To design data entry and input procedures
* To reduce input volume
* To design source documents for data capture or devise other data capture methods
* To design input data records, data entry screens, user interface screens, etc.
* To use validation checks and develop effective input controls.

**Output Design:**

The design of output is the most important task of any system. During output design, developers identify the type of outputs needed, and consider the necessary output controls and prototype report layouts.

### Objectives of Output Design:

The objectives of input design are:

* To develop output design that serves the intended purpose and eliminates the production of unwanted output.
* To develop the output design that meets the end user’s requirements.
* To deliver the appropriate quantity of output.
* To form the output in appropriate format and direct it to the right person.
* To make the output available on time for making good decisions.

**MODULES:**

1. **User**:
   1. **View Home page:**

Here user view the home page of the Disease application.

* 1. **View about page:**

In the about page, users can learn more about the Disease platform.

* 1. **Diabetes:**

User will predict the diabetes disease.

* 1. **Breast Cancer:**

User will predict the cancer disease.

* 1. **Liver:**

User will predict the liver disease.

* 1. **Kidney:**

User will predict the kidney disease.

* 1. **Skin**:

User will upload the image.

1. **System**
   1. **Working on dataset:**

System checks for data whether it is available or not and load the data in csv files.

* 1. **Pre-processing:**

Data need to be pre-processed according the models it helps to increase the accuracy of the model and better information about the data.

* 1. **Training the data:**

After pre-processing the data will split into two parts as train and test data before training with the given algorithms.

* 1. **Model Building**

To create a model that predicts the personality with better accuracy, this module will help user.

* 1. **Generated Score:**
  2. Here user view the score in %
  3. **Generate Results:**

We train the machine learning algorithm and predict the multi- diseases.

**UML DIAGRAMS**

UML stands for Unified Modelling Language. UML is a standardized general-purpose modelling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object-oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modelling Language is a standard language for specifying, Visualization, Constructing and documenting the artefacts of software system, as well as for business modelling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modelling of large and complex systems.

The UML is a very important part of developing objects-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

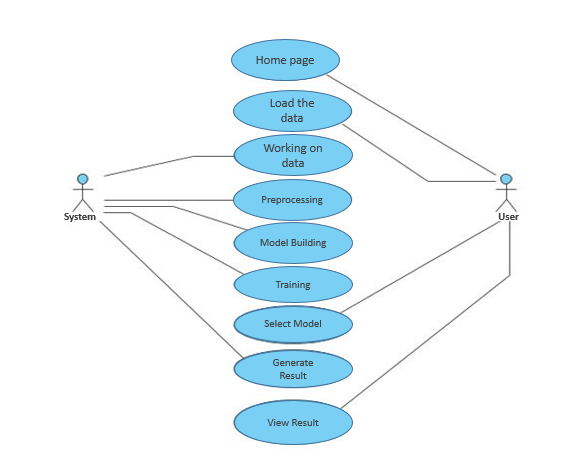
**GOALS:**

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modelling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modelling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

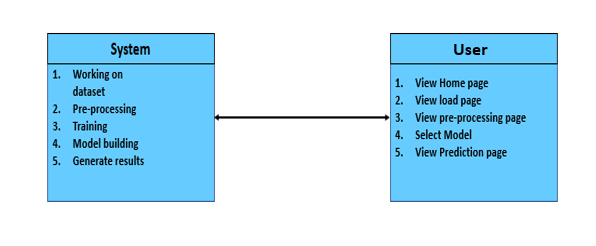
**USE CASE DIAGRAM**

* A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis.
* Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases.
* The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

****

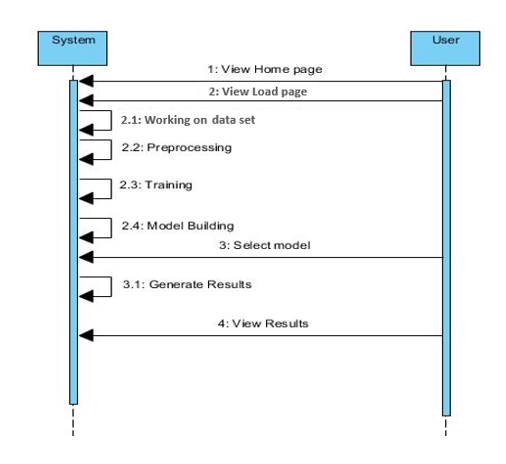
**CLASS DIAGRAM**

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information



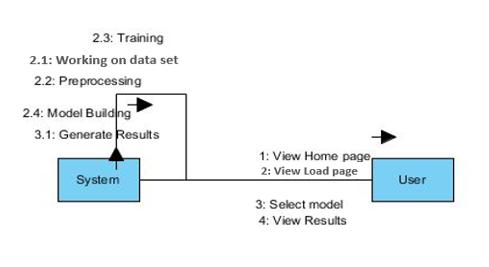
**SEQUENCE DIAGRAM**

* A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order.
* It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams



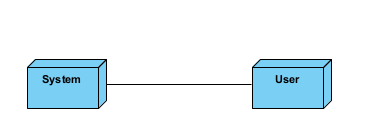
**COLLABORATION DIAGRAM:**

In collaboration diagram the method call sequence is indicated by some numbering technique as shown below. The number indicates how the methods are called one after another. We have taken the same order management system to describe the collaboration diagram. The method calls are similar to that of a sequence diagram. But the difference is that the sequence diagram does not describe the object organization whereas the collaboration diagram shows the object organization.



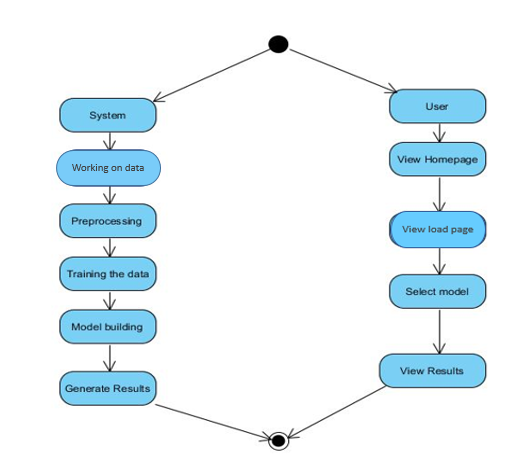
**DEPLOYMENT DIAGRAM**

Deployment diagram represents the deployment view of a system. It is related to the component diagram. Because the components are deployed using the deployment diagrams. A deployment diagram consists of nodes. Nodes are nothing but physical hardware’s used to deploy the application.



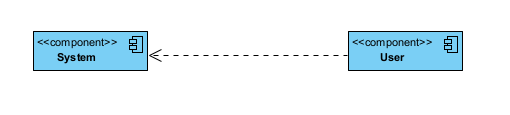
**ACTIVITY DIAGRAM:**

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modelling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.



**COMPONENT DIAGRAM**:

A component diagram, also known as a UML component diagram, describes the organization and wiring of the physical **c**omponents in a system. Component diagrams are often drawn to help model implementation details and double-check that every aspect of the system's required function is covered by planned development.



**ER DIAGRAM:**

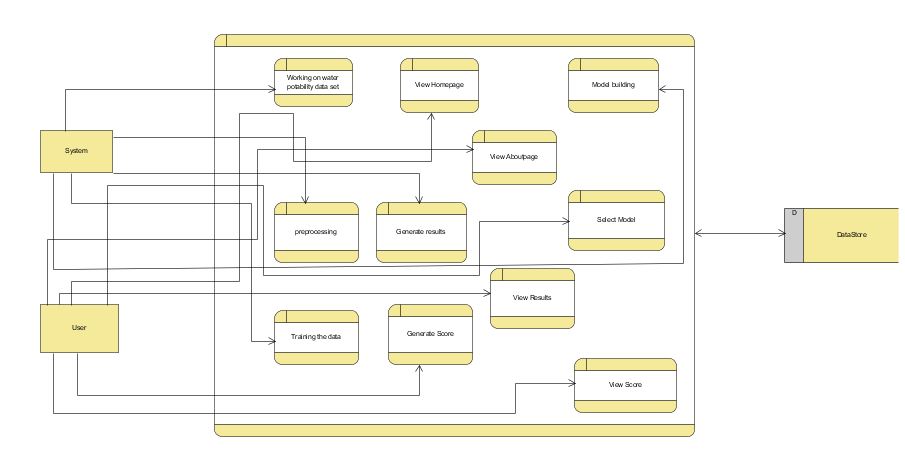
An Entity–relationship model (ER model) describes the structure of a database with the help of a diagram, which is known as Entity Relationship Diagram (ER Diagram). An ER model is a design or blueprint of a database that can later be implemented as a database. The main components of E-R model are: entity set and relationship set.

An ER diagram shows the relationship among entity sets. An entity set is a group of similar entities and these entities can have attributes. In terms of DBMS, an entity is a table or attribute of a table in database, so by showing relationship among tables and their attributes, ER diagram shows the complete logical structure of a database. Let’s have a look at a simple ER diagram to understand this concept.

# 

**DFD DIAGRAM:**

A Data Flow Diagram (DFD) is a traditional way to visualize the information flows within a system. A neat and clear DFD can depict a good amount of the system requirements graphically. It can be manual, automated, or a combination of both. It shows how information enters and leaves the system, what changes the information and where information is stored. The purpose of a DFD is to show the scope and boundaries of a system as a whole. It may be used as a communications tool between a systems analyst and any person who plays a part in the system that acts as the starting point for redesigning a system.



**CONCLUSION:**

Multiple disease detection using advanced machine learning techniques and diagnostic tools is a promising approach to improve patient outcomes and reduce healthcare costs. The early detection of diabetes, chronic kidney disease, liver disease, and breast cancer can enable timely interventions and improve patient survival rates. The integration of medical data, including patient history, clinical and laboratory tests, and imaging results, can enhance disease diagnosis and risk assessment. However, the accuracy and reliability of disease detection models depend on the quality and quantity of data used for training, as well as the validation and generalization of these models across different patient populations. Therefore, further research is needed to optimize the performance and implementation of multiple disease detection systems in clinical settings.

**FUTURE ENHANCEMENT**

Future enhancements for this multiple disease detection system involve continuous refinement and expansion. Firstly, the inclusion of additional diseases and conditions for detection, broadening its scope and impact. Secondly, the integration of real-time data streams and wearable device inputs to enable continuous monitoring. Thirdly, the incorporation of explainable AI techniques for greater transparency and trust in the system's predictions. Moreover, the system can benefit from interoperability with other healthcare systems, facilitating seamless data sharing and collaboration among healthcare providers. Lastly, ongoing research and development efforts will focus on improving the system's accuracy, scalability, and adaptability to emerging medical trends and technologies.

**REFERENCES:**

[1] J. L. Scully, ‘‘What is a disease?’’ EMBO Rep., vol. 5, no. 7, pp. 650–653, 2004.

[2] R. Leaman, R. Islamaj Dogan, and Z. Lu, ‘‘DNorm: Disease name normalization with pairwise learning to rank,’’ Bioinformatics, vol. 29, no. 22, pp. 2909–2917, Nov. 2013.

[3] N. Armstrong and P. Hilton, ‘‘Doing diagnosis: Whether and how clinicians use a diagnostic tool of uncertain clinical utility,’’ Social Sci. Med., vol. 120, pp. 208–214, Nov. 2014.

[4] A.-L. Barabási, N. Gulbahce, and J. Loscalzo, ‘‘Network medicine: A network-based approach to human disease,’’ Nature Rev. Genet., vol. 12, no. 1, pp. 56–68, Jan. 2011.

[5] R. H. Scheuermann, W. Ceusters, and B. Smith, ‘‘Toward an ontological treatment of disease and diagnosis,’’ Summit Transl. Bioinformat., vol. 2009, p. 116, Mar. 2009.

[6] P. Croft, D. G. Altman, and J. J. Deeks, ‘‘The science of clinical practice: Disease diagnosis or patient prognosis? Evidence about ‘what is likely to happen’ should shape clinical practice,’’ BMC Med., vol. 13, no. 1, p. 20, 2015.

[7] E. Choi, M. T. Bahadori, A. Schuetz, W. F. Stewart, and J. Sun, ‘‘Doctor ai: Predicting clinical events via recurrent neural networks,’’ in Proc. Mach. Learn. Healthcare Conf., 2016, pp. 301–318.

[8] C. C. Lee, ‘‘Fuzzy logic in control systems: Fuzzy logic controller. I,’’IEEE Trans. Syst., Man, Cybern., vol. 20, no. 2, pp. 404–418, Mar./Apr. 1990.

[9] J. Yen and R. Langari, Fuzzy Logic: Intelligence, Control, and Information, vol. 1. Upper Saddle River, NJ, USA: Prentice-Hall, 1999.

[10] H. D. Beale, H. B. Demuth, and M. Hagan, Neural Network Design. Boston, MA, USA: PWS, 1996.