A Vision for Better Health: HealthCare's 6-in-1 Medical Project for Disease Detection by Using Machine Learning and Deep Learning

Submitted in partial fulfillment of the requirements for the degree of

BACHELOR OF TECHNOLOGY
in
CSE (Data Science)

By

P.SWAMY	**20K91A6730**
G.MANVITHA	**20K91A6707**
J.SWATHI	**20K91A6714**
B.LIKITH	**20K91A6705**
K.DAVID	**20K91A6719**

Under the guidance of

MRS.YASMEEN SHABNUM
Assistance Prof. of CSE (Data Science)

DEPARTMENT OF COMPUTER SCIENCEAND ENGINEERING
(DATA SCIENCE)
TKR COLLEGE OF ENGINEERING AND TECHNOLOGY
(AUTONOMOUS)
(ACCREDITED BY NBA AND NAAC WITH 'A+' GRADE)
Medbowli, Meerpet, Saroornagar, Hyderabad-500097

DECLARATION BY THE CANDIDATES

The work presented in this Major Project phase I entitled "**A Vision for Better Health: HealthCare's 6-in-1 Medical Project for Disease Detection by Using Machine Learning and Deep Learning"** is original and has been carried out under the supervision of **MRS.YASMEEN SHABNUM Assistance Professor**, in Department of Computer Science and Engineering (Data Science) is submitted in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering (Data Science).

P. SWAMY	(20K91A6730)
G.MANVITHA	(20K91A6707)
J.SWATHI	(20K91A6714)
B.LIKITH	(20K91A6705)
K.DAVID	(20K91A6719)

CERTIFICATE

This is to certify that the Major Project Phase I project report entitled **A Vision for Better Health: HealthCare's 6-in-1 Medical Project for Disease Detection by Using MachineLearning and DeepLearning**, is a bonafide work done by **"P.SWAMY (20K95A6730) , G.MANVITHA (20K91A6707) , J.SWATHI (20K91A6714) , B.LIKIT (20K91A6705), K.DAVID (20K91A6719)** in partial fulfillment of the academic requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering (Data Science) submitted to the Department of Computer Science and Engineering (Data Science), TKR College of Engineering and Technology, Hyderabad during the academic year 2023-2024.

Signature of the Guide
MRS.Yaseem Shabnum
Assistance Prof.

Signature of the HoD
Dr.V.Krishna
Professor

Signature of the Co-ordinator
Mr.M.Arokia Muthu
Asst.Prof

Signature of the External

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompanies the successful completion of any task would be incomplete without the mention of the people who made it possible and whose encouragement and guidance have crowned our efforts with success.

We express our sincere gratitude to **Management of TKRCET** for granting permission and giving inspiration for the completion of the project work.

Our faithful thanks to our **Principal Dr. D. V. Ravi Shankar, MTech., Ph.D.,** TKR College of Engineering & Technology for his Motivation in studies and completion of the project work.

Our gratitude also indebted to the **Head of the Department**, **Dr. V. Krishna**, Computer Science and Engineering (Data Science), TKR College of Engineering and Technology, for his support and guidance throughout our Dissertation.

Thanks to our Project Coordinator **Mr.M.Arokia Muthu M.E.,(Ph.D.),** Assistant Professor, Department of CSE (Data Science), TKR College of Engineering & Technology for his constant encouragement.

As a team, we are indebted to our Internal Guide**, MRS.YASMEEN SHABNUM**, Assistance Professor, Dept. of Computer Science and Engineering (Data Science), TKR College of Engineering and Technology, for her support and guidance throughout our Dissertation.

Finally, we want to thanks each team member and everyone else who played a role in successfully completing this Dissertation. Furthermore, We would like to thank my family and friends for their moral support and encouragement

By,

P. SWAMY	20K91A6730
G.MANVITHA	20K91A6707
J.SWATHI	20K91A6714
B.LIKITH	20K91A6705
K.DAVID	20K91A6719

TABLE OF CONTENTS

ABSTRACT

This project addresses the prevalent health challenges of our time by consolidating seven disease detection modules into a unified platform, empowered by Artificial Intelligence (AI). This innovative solution enables users to effortlessly obtain rapid test results in the comfort of their homes with just a few clicks. Leveraging Python, Anaconda, TensorFlow-gpu, Flask, and OpenCV, our software employs Convolutional Neural Networks (CNNs) to efficiently process images and recognize patterns, inspired by the connectivity of neurons in the human brain. Covering diseases ranging from Covid-19 to heart conditions, each module is tailored to detect specific symptoms. While the experimental nature of our results is acknowledged, the project holds promise for a transformative impact on healthcare. Future plans involve expanding disease detections, enhancing data enrichment, educating users on precautions and self-care, and implementing a record storage feature. This initiative aligns with the evolving landscape of AI-driven healthcare, aiming to contribute to improved health outcomes globally.

The diseases covered in our project, ranging from infectious diseases like Covid-19 to chronic conditions such as Alzheimer's and heart disease, showcase the versatility of our approach. We acknowledge the experimental nature of our results while recognizing the potential for significant advancements in healthcare.

Looking ahead, our roadmap includes continuous refinement and expansion. We plan to incorporate additional disease detection modules, broadening the range of detectable conditions through methods like X-ray scans or numerical inputs. Furthermore, we aim to enhance our models by continuously training them on diverse and extensive datasets. User education is a key focus, with plans to integrate features that provide guidance on precautions and self-care measures for individuals who receive positive test results. Additionally, a record storage feature is in the pipeline to facilitate easy access to historical detection records for both users and healthcare professionals.

LIST OF FIGURES

LIST OF TABLES

CHAPTER-01

INTRODUCTION

1. Motivation

In this project is fueled by a strong motivation to redefine healthcare by leveraging the transformative capabilities of Artificial Intelligence (AI). With a deep commitment to making healthcare accessible to all, our project seeks to break down geographical and socioeconomic barriers. We are inspired by the belief that timely access to critical healthcare information can lead to early interventions, potentially mitigating the severity and impact of various diseases. The democratization of diagnostics is a driving force, as we aim to bring advanced medical technologies to a broader audience, irrespective of their resources or location. Encompassing a diverse range of diseases, from infectious to chronic conditions, our project is motivated by a desire to provide a holistic approach to disease detection and management. The innovative use of Convolutional Neural Networks represents our dedication to pushing the boundaries of AI for enhanced accuracy in disease detection. Central to our motivation is the empowerment of individuals through immediate test results and relevant information, enabling them to take proactive control of their health. Our commitment extends to continuous improvement, with plans to refine and expand our platform over time based on accumulated data and user feedback. In essence, our project aspires to contribute to a future where healthcare is not only technologically advanced but also inclusive, timely, and empowering for individuals worldwide.

2. Problem definition:

A Vision for Better Health, HealthCare's 6-in-1 Medical Project for Disease Detection to develop AI-powered Online Medical Diagnostics Platform which revolutionize medical science with an integrated platform using AI and ML for rapid disease detection with high accuracy only on a single application rather than the existing systems that uses different applications for different disease detections.

3. Limitations of existing system:

Existing healthcare systems, scattered across platforms and disease-specific, face accuracy issues with outdated technologies and limited coverage. The use of obsolete algorithms contributes to compromised performance. Complex user interfaces hinder accessibility, and some systems lack optimal response times. Our project addresses these challenges by developing a unified platform for six diseases.

Proposed System:

The proposed system aims to unify disease detection by consolidating six illnesses onto a single platform, eliminating the complexities of current fragmented systems. Employing advanced algorithms, we prioritize high accuracy within a short processing time, addressing the inefficiencies of outdated technologies. Our user-friendly interface ensures seamless navigation, while optimizing response time for swift results. Enabling at-home disease detection, our system promotes early identification and intervention. Affordable pricing ensures accessibility for users of diverse economic backgrounds. Leveraging sophisticated image analysis algorithms, including those for X-ray and MRI scans, enhances the system's diagnostic capabilities. This holistic approach positions our system as a groundbreaking solution for precise and efficient healthcare diagnostics.

CHAPTER-02

LITERATE REVIEW

1. Review of Literature:

The purpose of a literature review is to understand existing research and debates in a specific area and present this knowledge in a written report. It helps build knowledge in a field, understanding concepts, methods, and techniques used. Literature reviews also teach how research findings are presented in the discipline, aiding in successful writing. They show the state of research and identify gaps, guiding new research. Researchers conduct reviews to find areas lacking detailed research, contributing to knowledge development. Literature reviews should not just summarize sources but compare, contrast, and critically evaluate them, showing their contribution to the topic's body of knowledge.

PAPER-1: Recursion Enhanced Random Forest With an Improved Linear Model (RERF-ILM) for Heart Disease Detection on the Internet of Medical Things Platform.

The study conducted by Chunyan Guo et al. involved researchers from the Anesthesiology Department at The Affiliated Hospital of Inner Mongolia Medical University, Hohhot, China, and the Graduate School of Medical School of Chinese PLA Hospital, Beijing, China. The corresponding author, Jianshe Yu, can be contacted via email at yujianshe@nmgfy.com. Their work was supported by Inner Mongolia Medical University through the ‘‘One Million Yuan Project of Science and Technology’’ under Grant YKD2018KJBW(LH)022.

Merits:

1. **High Accuracy**: The proposed RERF-ILM method achieves high accuracy in heart disease prediction, as indicated by its precision through the heart disease prediction model.
2. **Integration of Models**: It merges the features of the linear model and random forest, which can provide complementary strengths and enhance overall performance.
3. **Utilization of Machine Learning**: Leveraging machine learning techniques allows for the processing of raw health data and offers new insights into heart disease prediction, potentially leading to better patient outcomes.

Demerits:

1. **Complexity**: The algorithm described, incorporating a combination of models and techniques, may be complex to implement and understand, requiring expertise in both machine learning and healthcare domain knowledge.
2. **Data Dependency**: The effectiveness of the RERF-ILM method relies heavily on the quality and quantity of data available for training the model. Insufficient or biased data may lead to inaccurate predictions.
3. **Scalability**: While the method may perform well in research settings or with limited datasets, its scalability to larger populations or real-world deployment scenarios needs to be evaluated.

Paper-2: Brain Tumour Detection Based on Multimodal Information Fusion and Convolutional Neural Network

The paper titled "Brain Tumor Detection Based on Multimodal Information Fusion and Convolutional Neural Network" authored by Ming Li, Lishan Kuang, Shuhua Xu, and Zhanguo Sha presents a novel approach to address the challenge of low accuracy in traditional brain tumor detection. Referred to as multi-CNNs, their method combines multimodal information fusion with convolutional neural networks (CNNs). By extending 2D-CNNs to multimodal 3D-CNNs, the model effectively captures brain lesions across different modal characteristics in three-dimensional space, improving upon the limitations of 2D-CNNs. Addition of real normalization layers between convolution and pooling layers enhances network convergence and mitigates overfitting issues. Furthermore, the authors propose an improved loss function incorporating weighted loss to enhance feature learning of lesion areas.

Merits:

1. **Integration of Multimodal Information:** The proposed algorithm effectively combines information from multiple MRI modalities, enhancing the accuracy and comprehensiveness of brain tumor detection.
2. **Extension to 3D-CNNs:** By extending 2D-CNNs to multimodal 3D-CNNs, the algorithm captures three-dimensional features of brain tumor lesions, improving upon traditional methods that rely on 2D information.
3. **Improved Loss Function:** Introduction of a weighted loss function reduces interference from non-lesional areas, enhancing the accuracy of tumor detection.
4. **Review of Existing Methods:** The paper provides a comprehensive review of existing brain tumor detection methods, offering valuable insights into their strengths and limitations.

5. **Application of CNNs:** Leveraging CNNs for brain tumor detection reflects the state-of-the-art in deep learning techniques and their potential to revolutionize medical image analysis.

Demerits:

1. **Complexity:** The proposed algorithm may involve complex computational processes, potentially requiring substantial computational resources for implementation and training.
2. **Data Requirements:** CNN-based methods often rely on large volumes of labeled training data, which may be challenging to obtain, particularly for medical imaging datasets.
3. **Interpretability:** Deep learning models like CNNs are often considered "black boxes," making it difficult to interpret their decisions, which could be a concern in critical medical applications where interpretability is crucial.
4. **Generalization:** The algorithm's performance may vary across different datasets and clinical settings, raising concerns about its generalizability beyond the specific conditions under which it was developed.

Paper-3: CXR-Net: A Multitask Deep Learning Network for Explainable and Accurate Diagnosis of COVID-19 Pneumonia From Chest X-Ray Images.

The paper titled "CXR-Net: A Multitask Deep Learning Network for Explainable and Accurate Diagnosis of COVID-19 Pneumonia From Chest X-Ray Images" presents a novel deep learning network aimed at accurately diagnosing COVID-19 pneumonia from chest X-ray images. Authored by Xin Zhang, Liangxiu Han, Tam Sobeih, Lianghao Han, Nina Dempsey, Symeon Lechareas, Ascanio Tridente, Haoming Chen, Stephen White, and Daoqiang Zhang, the research focuses on developing a multitask network capable of both accurate diagnosis and explainable results, addressing the critical need for reliable COVID-19 detection methods. The proposed CXR-Net leverages deep learning techniques to achieve high diagnostic accuracy while providing insights into the decision-making process, making it potentially valuable in clinical settings.

Merits:

1. **Accurate Diagnosis**: CXR-Net aims to provide accurate diagnosis of COVID-19 pneumonia from chest X-ray images, which is crucial for effective patient treatment.
2. **Interpretability**: Unlike many existing deep learning models, CXR-Net offers transparency and interpretability through pixel-level visual explanations. This feature can enhance clinicians' understanding of the model's predictions.

3. **Enhanced Visualization**: The proposed method generates high-resolution activation maps, providing detailed visual explanations for classification results. This can assist radiologists in identifying relevant areas associated with COVID-19 pneumonia.
4. **Generalizability**: CXR-Net is evaluated on diverse datasets from public and private sources, demonstrating its potential for generalization across different patient populations and imaging conditions.

Demerits:

1. **Complexity**: The architecture of CXR-Net, with its Encoder-Decoder-Encoder structure, may introduce complexity in implementation and training, requiring significant computational resources and expertise.
2. **Data Dependency**: The performance of CXR-Net is highly dependent on the quality and quantity of available training data. Limited or biased datasets may affect the model's accuracy and generalizability.
3. **Model Interpretation**: While CXR-Net offers visual explanations for its predictions, interpreting these explanations may still require expertise in radiology and deep learning, limiting its accessibility to non-experts.
4. **Ethical Considerations**: The deployment of AI-based diagnostic tools in clinical practice raises ethical concerns regarding patient privacy, algorithm bias, and accountability.

Paper-4: A Novel Hybrid K-Means and GMM Machine Learning Model for Breast Cancer Detection

A novel hybrid K-Means and Gaussian Mixture Model (GMM) machine learning model for breast cancer detection is proposed in this study. Led by P. Esther Jebarani and N. Umadevi from the Department of Computer Science at Sri Jayendra Saraswathy Maha Vidyalaya College of Arts and Science in Coimbatore, India, and Hien Dang, a member of IEEE, from the Faculty of Computer Science and Engineering at Thuyloi University in Hanoi, Vietnam, and the Department of Computer Science at the University of Massachusetts Boston, USA. The corresponding author is Hien Dang (hiendt@tlu.edu.vn). This research.

Methodology demonstrates enhanced performance in breast cancer detection. By introducing a novel parameter for evaluating the efficacy of K-means and Gaussian mixture models, this research contributes significantly to the field. The hybrid segmentation and detection approach presented offer promise in accurately classifying benign and malignant tumors.

Merits:

- The proposed method integrates a hybrid technique for breast cancer detection, combining K-means segmentation and Gaussian mixture model (GMM) machine learning.
- By segmenting the disordered portion of cancerous cells in breast images, the method aims to improve early detection and increase treatment options.

Demerits:

- Despite advancements, automated systems for breast cancer detection still face challenges in accurately distinguishing between benign and malignant tumors.
- The reliance on machine learning models and segmentation techniques may introduce complexities and computational overhead.
- There could be limitations in the dataset used for training and validation, potentially affecting the generalizability of the proposed method.

Literature Review Conclusion:

The system's limitations encompass a spectrum of challenges, spanning from reduced accuracy and longer response times to increased expenses and a lack of user-friendliness. Each of these factors exerts its own influence on the system's functionality and user experience, ultimately impacting its overall effectiveness. Reduced accuracy undermines the system's reliability, casting doubt on the trustworthiness of its outputs and potentially leading to misinformed decisions. Longer response times impede efficiency, slowing down processes and hindering timely actions or interventions. Increased expenses strain resources, potentially making the system less accessible or affordable for users. Moreover, a lack of user-friendliness complicates interactions, creating barriers to adoption and hindering the system's usability.

These limitations, if left unaddressed, can have far-reaching implications, compromising the system's reliability, efficiency, affordability, and ease of use. Therefore, it is imperative to tackle these challenges head-on to enhance the system's performance and usability. Strategies such as refining algorithms to improve accuracy, optimizing infrastructure to minimize response times, implementing cost-effective solutions to mitigate expenses, and enhancing user interfaces to improve user-friendliness can all contribute to overcoming these limitations. By addressing these issues comprehensively, the system can realize its full potential, delivering more reliable, efficient, affordable, and user-friendly solutions that better serve its intended purpose and user base.

CHAPTER-03

REQUIREMENTSANALYSIS

Requirements analysis is the process of identifying, documenting, and evaluating the needs and expectations of stakeholders for a software system or solution. It involves understanding what the system should do, how it should behave, and what qualities it should possess in order to meet the goals and objectives of the stakeholders. This process is essential for ensuring that the final product meets the desired outcomes and provides value to its users.

In the context of the provided requirements analysis, it encompasses both functional and non-functional requirements. Functional requirements define specific functionalities and features that the system should have, such as disease detection, test result delivery, precautionary information display, data storage, continuous improvement, user education, mobile compatibility, and security measures.

On the other hand, non-functional requirements specify the qualities and characteristics of the system beyond its specific functionalities. These include aspects such as performance, reliability, scalability, usability, security, compatibility, maintainability, accuracy and precision, privacy, feedback and improvement, training and support, and cost-effectiveness. These non-functional requirements ensure that the system is not only functional but also efficient, reliable, secure, user-friendly, and adaptable to changing needs and circumstances.

Overall, requirements analysis is crucial for guiding the design, development, and implementation of a software system, ensuring that it meets the needs and expectations of its stakeholders and delivers value to its users.

3.1 Functional Requirements:

Disease Detection Module:

The system shall include an AI-driven disease detection module utilizing Convolutional Neural Networks (CNN) for the detection of multiple diseases including Covid-19, Brain Tumor, Breast Cancer, Alzheimer's Disease, Diabetes, and Heart Disease. Continuous learning mechanisms shall be implemented to ensure ongoing improvement through the incorporation of new data.

Test Result Delivery:

A user-friendly interface shall be provided to enable users to upload test data, including images and numerical inputs. The system shall conduct AI analysis on uploaded data and present immediate results in a clear format, facilitating easy interpretation by users.

Precautionary Information:

For users receiving positive results, the system shall display precautionary measures and recommendations tailored to each specific disease. Additionally, educational content related to disease management shall be provided to assist users in understanding and addressing their health concerns.

Data Storage:

To maintain anonymity and ensure security, the system shall establish a secure data storage system for the anonymous storage of detection records.

Additional Diseases and Features (Future Planning):

In anticipation of future expansions, the system shall be designed to integrate additional diseases detectable via X-ray scans or numerical inputs. Moreover, consideration shall be given to adding features such as displaying precautions and potential treatment options for detected diseases.

Continuous Improvement:

A systematic approach shall be implemented to facilitate updates to the AI model based on new data, research findings, and technological advancements, ensuring the system's effectiveness and relevance over time.

User Education:

A dedicated section shall be developed within the system to educate users about various diseases, their symptoms, and the importance of early detection, empowering users to make informed decisions about their health.

Mobile Compatibility:

The system shall be designed to ensure platform accessibility across various devices, including mobile phones and tablets, enabling users to access its functionalities conveniently.

Security Measures:

Robust security measures shall be implemented to safeguard user data, including strong encryption for data in transit and at rest. Regular updates to security protocols shall be conducted to address emerging threats and ensure compliance with relevant regulations.

3.2 Non-Functional Requirements:

Performance:

The system shall demonstrate quick and responsive disease detection capabilities, providing results within seconds. Furthermore, system availability shall be maintained 24/7 with minimal downtime to ensure uninterrupted access for users.

Reliability:

A high level of accuracy in disease detection shall be achieved, instilling confidence in users regarding the reliability of the platform. Additionally, the system shall be reliable and consistently available for use by users.

Scalability:

The system architecture shall be designed to accommodate a growing user base and expanding datasets. Scalability considerations shall also extend to future disease additions, ensuring the system's adaptability to evolving requirements.

Usability:

An intuitive and user-friendly interface shall be developed to enhance usability, catering to users with varying technical skills. Accessibility features shall be incorporated to facilitate seamless interaction with the system.

Security:

Stringent security measures shall be implemented to protect user data, including compliance with data protection and privacy regulations. Measures shall include encryption for data protection and adherence to best practices for data handling.

Compatibility:

The system shall ensure compatibility with popular browsers and operating systems, facilitating seamless access for users across different platforms. Additionally, the mobile version of the platform shall be optimized for various devices to enhance accessibility.

Maintainability:

The system architecture shall be designed for ease of maintenance, allowing for updates and modifications with minimal disruption. Comprehensive documentation shall be provided to support future development efforts and ensure system maintainability.

Accuracy and Precision:

Continuous efforts shall be made to maintain high accuracy and precision in disease detection. Regular validation and calibration of the AI model with diverse datasets shall be conducted to ensure reliable performance.

Privacy:

User privacy shall be prioritized through anonymization and secure storage of user data. Adherence to privacy regulations and obtaining explicit user consent shall be integral to the system's data handling processes.

Feedback and Improvement:

Mechanisms shall be established to collect regular user feedback for continuous improvement. Feedback shall be utilized to enhance functionality and user experience, ensuring the system meets user expectations effectively.

Training and Support:

User training materials shall be provided to facilitate user onboarding and familiarization with system functionalities. Responsive customer support services shall also be offered to address user queries and issues promptly.

Cost-Effectiveness:

Cost-effective solutions shall be implemented for system infrastructure and maintenance to optimize resource utilization and minimize operational costs, ensuring sustainable long-term viability.

CHAPTER-04

DESIGN

1. DFD, ER-Diagrams and System Architecture

1. DATAFLOW DIAGRAMS:

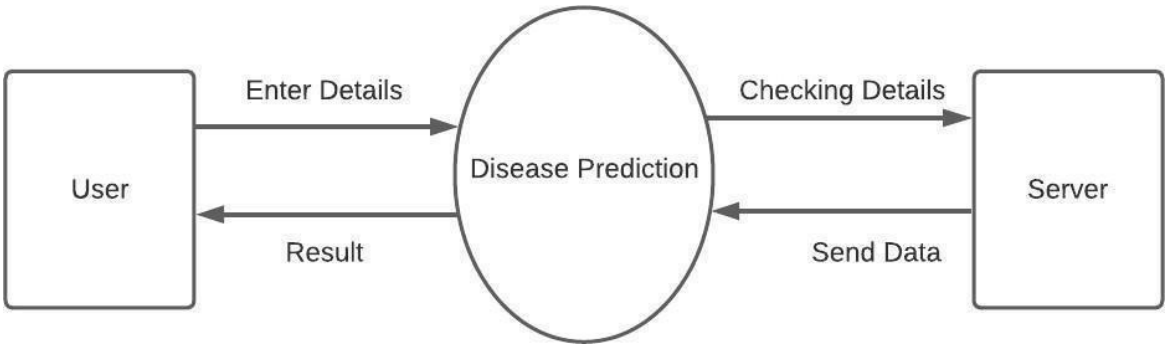

FIG.4.1.1 DATA FLOW DIAGRAM LEVEL 0

The system begins with "User Input," where users provide their test data for disease detection. This data is then processed through the "Disease Detection Module," which houses the core AI-powered process responsible for analyzing the input using a Convolutional Neural Network (CNN). The immediate test results are displayed to the user through the "Test Result Display." For positive disease detections, the system provides "Precautionary Information" to guide users on necessary actions. Additionally, the system securely stores anonymous detection records in the "Data Storage" for future reference and analysis.

FIRST DFD:

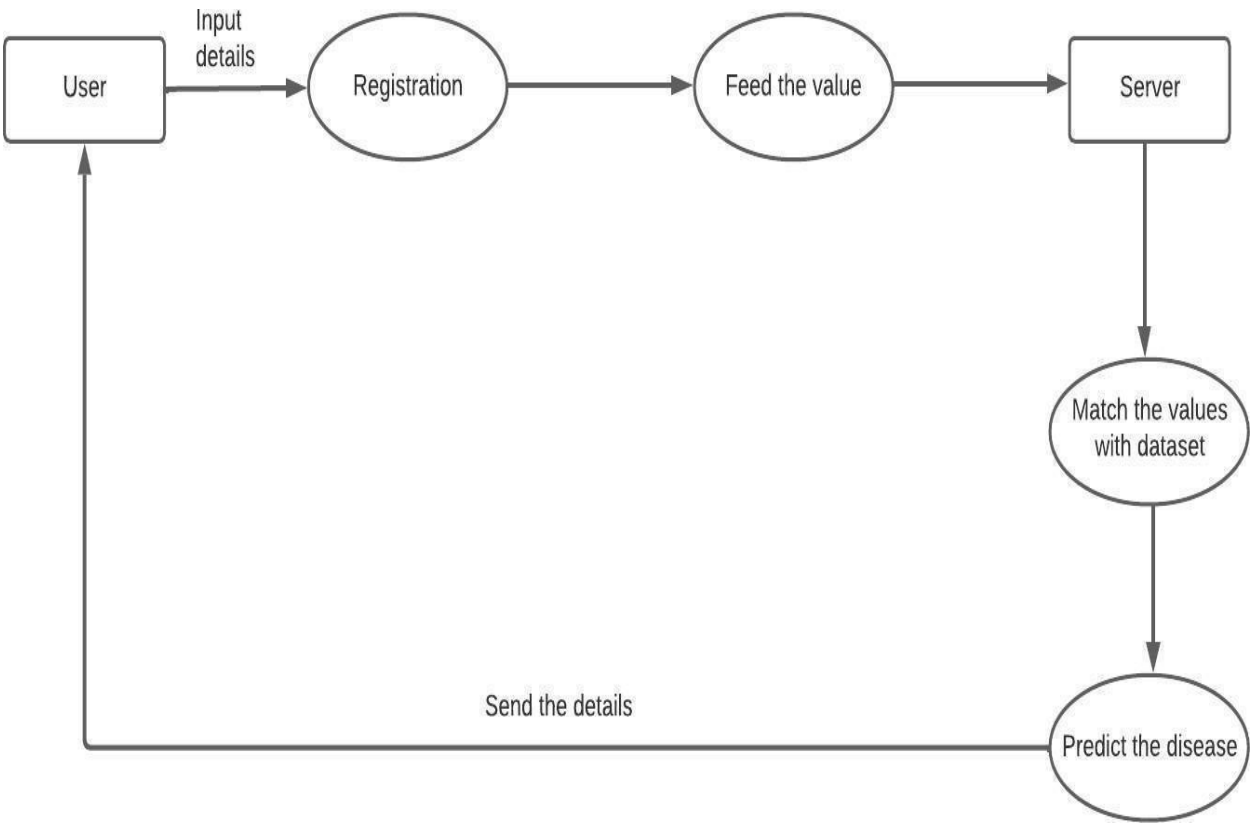

Fig. 4.1.2 First Level DFD

Within the "Disease Detection Module," the user input undergoes "Pre-Processing" to prepare the data for analysis. The core of the system lies in the "AI Disease Detection Model," utilizing a Convolutional Neural Network (CNN) to thoroughly analyze the input data and identify potential diseases. The output from the AI model is then processed in the "Result Processing" stage to generate clear and understandable formats. Finally, the processed results are displayed to the user through the "Test Result Display," completing the detailed view of the disease detection process.

SECOND LEVEL DFD

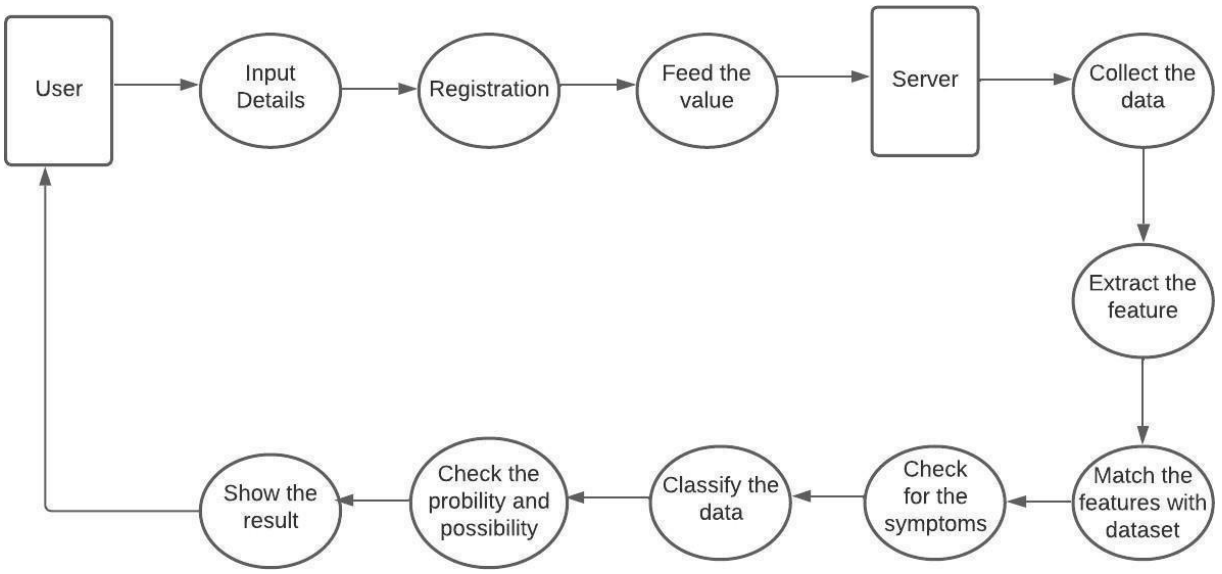

Fig. 4.1.3. Second Level DFD

In our disease detection system, the journey begins with users providing input data, including relevant test information. This data undergoes a meticulous pre-processing phase, where tasks such as image normalization and data validation are executed to optimize its readiness for analysis by the core component of our system—the AI Disease Detection Model.

At Level 2, this model is detailed to encompass specific steps like feature extraction and model training, ensuring its robust capability to accurately identify diseases. Following the AI analysis, the Result Processing stage refines the raw output into a clear and comprehensible format for user presentation. Users with positive results then receive personalized precautionary information, including disease-specific recommendations. The system's commitment to privacy is reflected in the Data Storage process at Level 2, emphasizing secure storage mechanisms, encryption, and indexing for anonymous and protected detection records. This comprehensive approach ensures not only accurate disease detection but also user-friendly result presentation and the safeguarding of sensitive information for future reference and analysis.

4.1.3System Architecture:

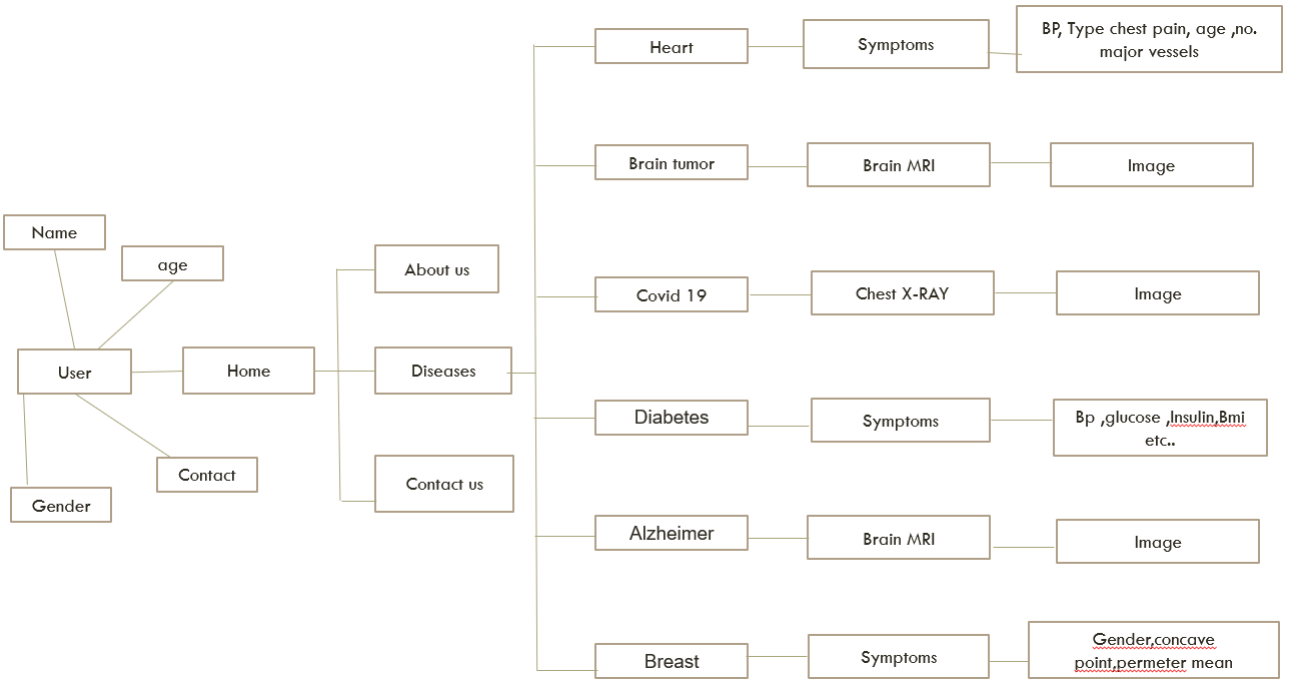

Fig. 4.1.4. System Architecture

Entity Relationship Table for Database Design diagram :

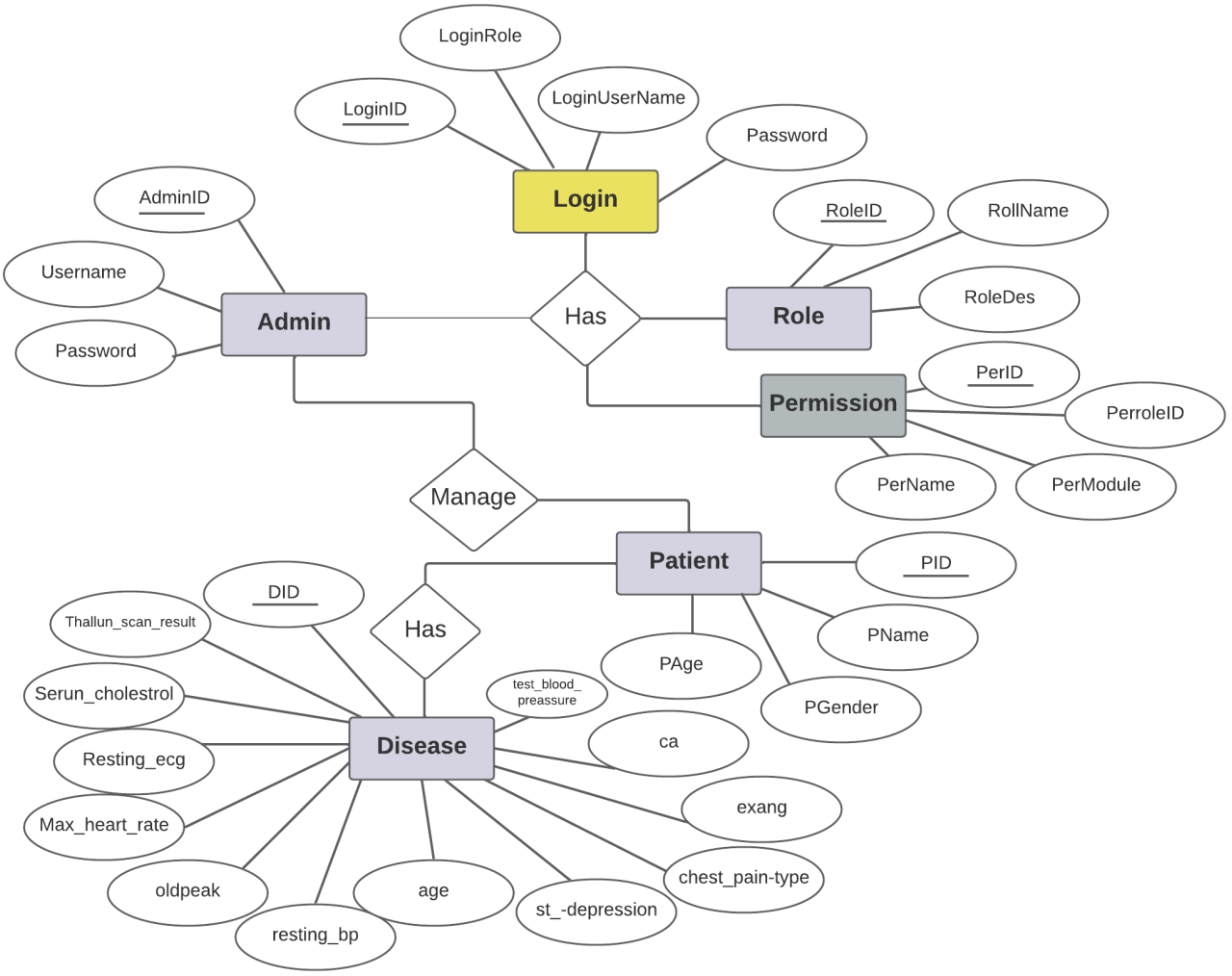

Fig. 4.1.5. ER Diagram

In our comprehensive health-focused project, we've designed an Entity-Relationship (ER) diagram to capture the core components of our disease detection platform. The primary entities include User, Disease, Test Result, Medical Image, Precaution, and Detection Record. Each entity is associated with specific attributes reflecting essential information. For instance, the User entity includes UserID, Username, Age, Gender, Address, and Contact Information. Relationships between entities have been established, such as the connection between User and Test Result, forming a one-to-many relationship where a user can have multiple test results. Similarly, connections between Disease and Test Result, Disease and Medical Image, Disease and Precaution, and User and Detection Record have been defined to represent the associations between these critical components of our healthcare system. This ER diagram provides a foundational structure for our database, ensuring efficient organization and retrieval of health-related data for accurate disease detection and analysis.

4.2 ALGORITHMS

CNN

Artificial Intelligence has come a long way and has been seamlessly bridging the gap between the potential of humans and machines. And data enthusiasts all around the globe work on numerous aspects of AI and turn visions into reality - and one such amazing area is the domain of Computer Vision. This field aims to enable and configure machines to view the world as humans do, and use the knowledge for several tasks and processes (such as Image Recognition, Image Analysis and Classification, and so on). And the advancements in Computer Vision with Deep Learning have been a considerable success, particularly with the Convolutional Neural Network algorithm.

Introduction to CNN

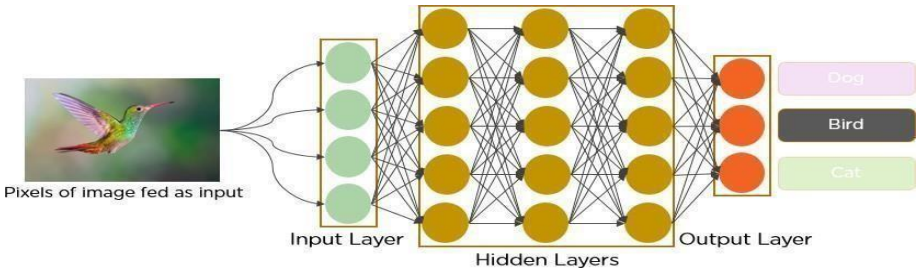

Fig 4.2.1 CNN making prediction on a bird Image

In CNN, every image is represented in the form of an array of pixel values.

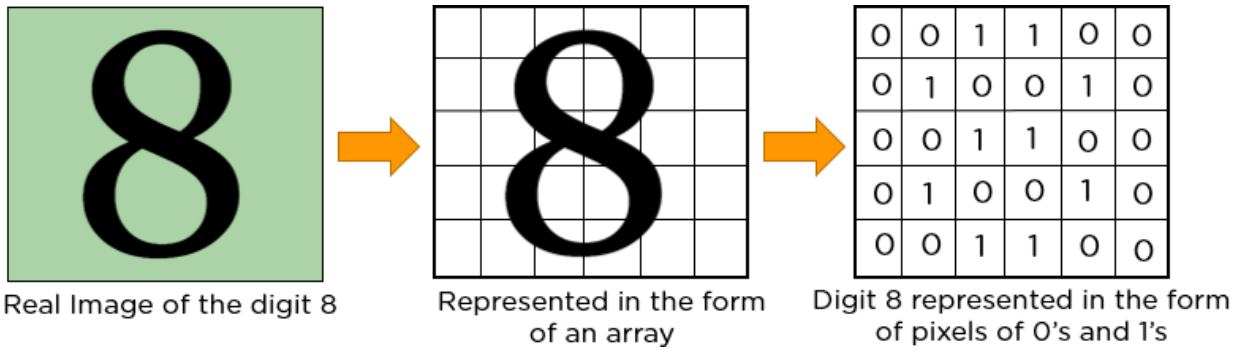

Fig 4.2.2 Representing 8 in pixels

Layers in a Convolutional Neural Network

A convolution neural network has multiple hidden layers that help in extracting information from an image. The four important layers in CNN are:

1. Convolution layer
2. ReLU layer
3. Pooling layer
4. Fully connected layer

Convolution Layer

This is the first step in the process of extracting valuable features from an image. A convolution layer has several filters that perform the convolution operation. Every image is considered as a matrix of pixel values.Consider the following 5x5 image whose pixel values are either 0 or 1. There's also a filter matrix with a dimension of 3x3. Slide the filter matrix over the image and compute the dot product to get the convolved feature matrix.

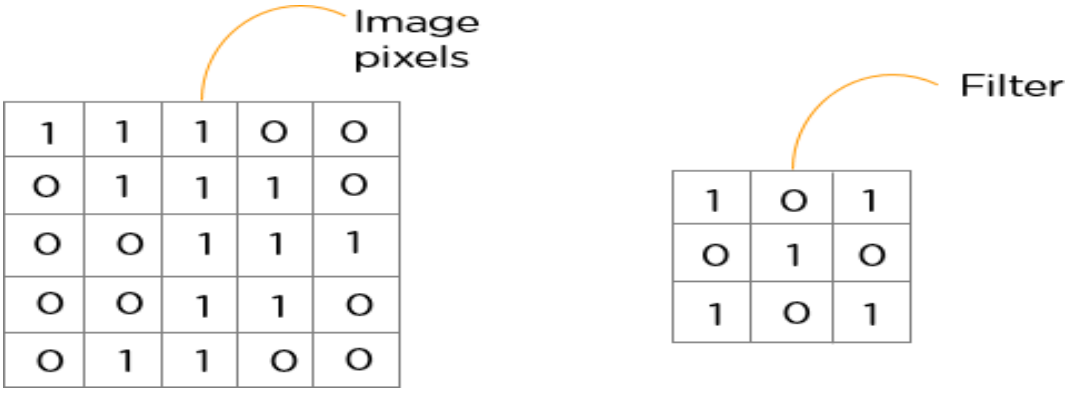

Fig .4.2.3. Showing image and filter

ReLU layer

ReLU stands for the rectified linear unit. Once the feature maps are extracted, the next step is to move them to a ReLU layer. ReLU performs an element-wise operation and sets all the negative pixels to 0.

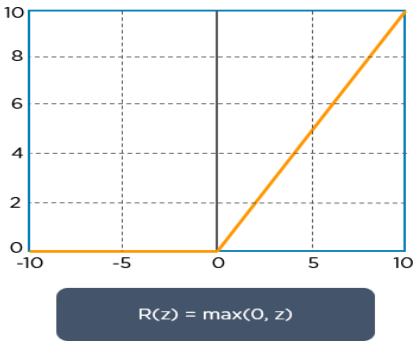

Fig.4.2.4. ReLU Activation Function

Pooling Layer:

Pooling is a down-sampling operation that reduces the dimensionality of the feature map. The rectified feature map now goes through a pooling layer to generate a pooled feature map.

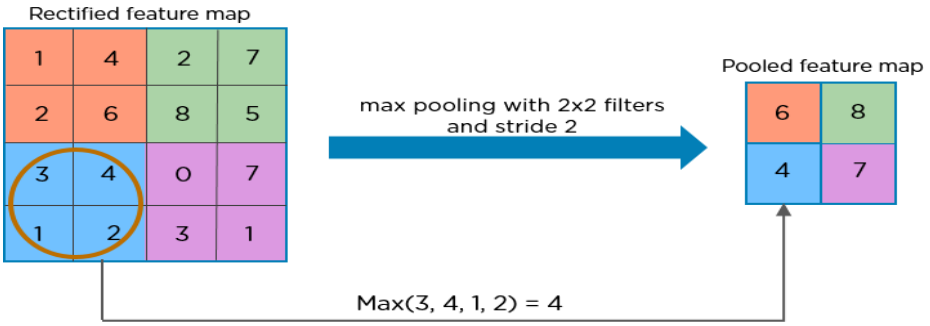

Fig 4.2.5. Max Pooling

The pooling layer uses various filters to identify different parts of the image like edges, corners, body, feathers, eyes, and beak.

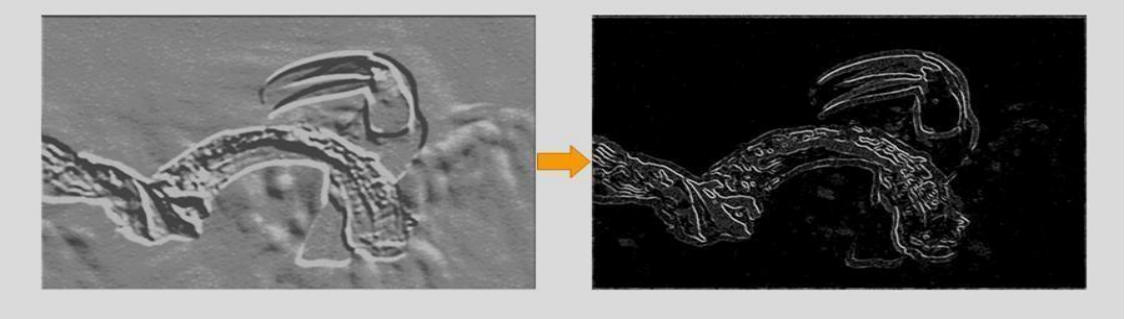

Fig .4.2.6 Max Pooled Image

Here's how the structure of the convolution neural network looks so far:

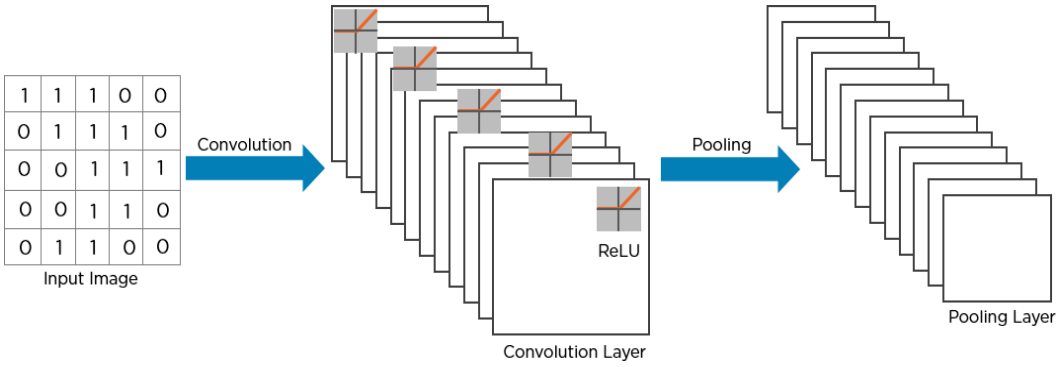

Fig 4.2.7. CNN Structure

The next step in the process is called flattening. Flattening is used to convert all the resultant 2-Dimensional arrays from pooled feature maps into a single long continuous linear vector.

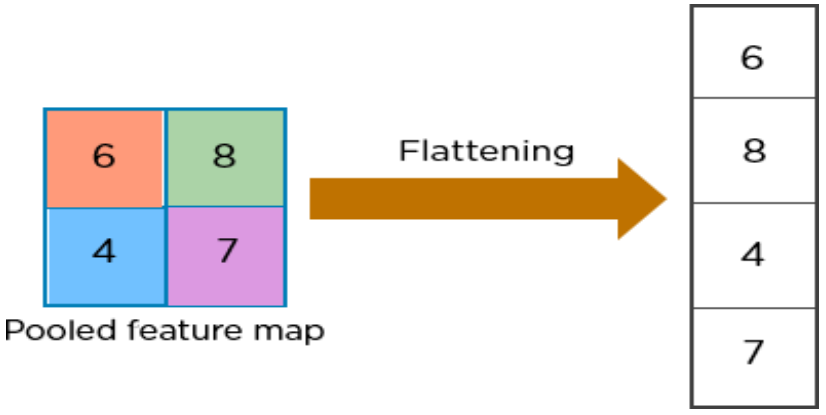

Fig 4.2.8. Flattening the Pooled Image

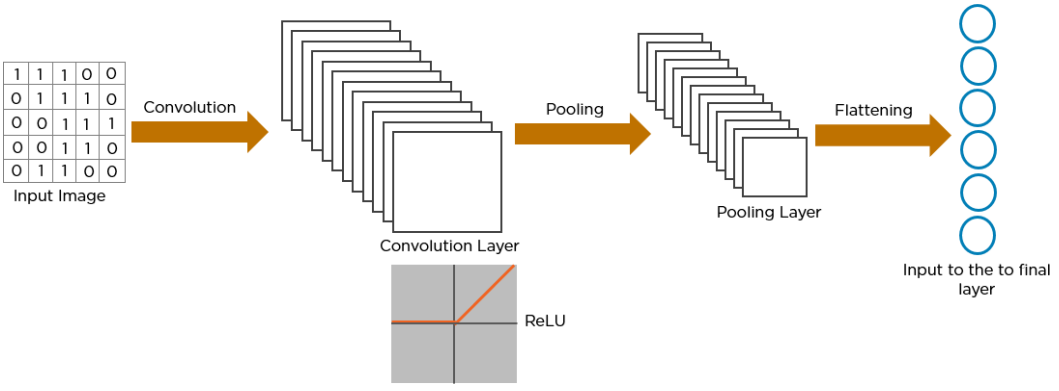

Fig 4.2.9. Flattened Image fed to Fully Connected Layer

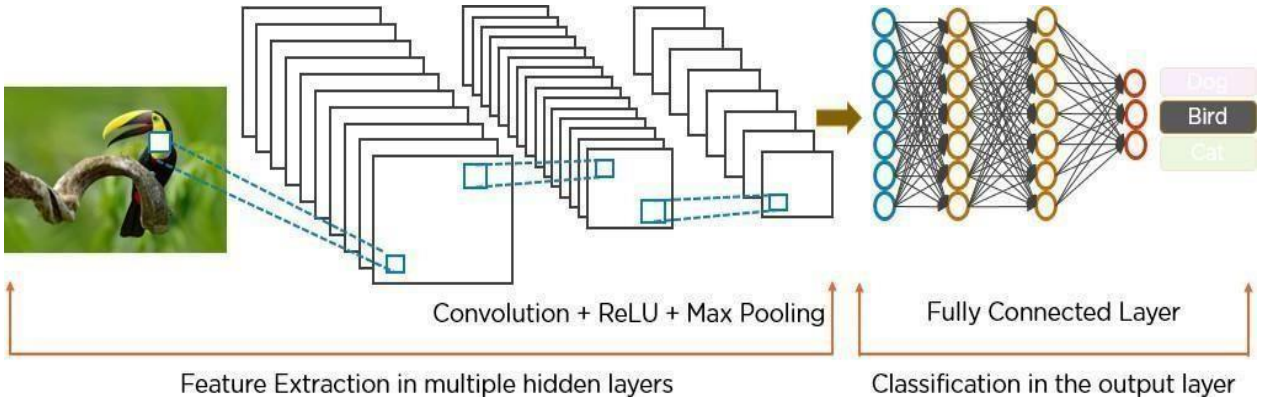

Fig 4.2.10 Two Different parts of a CNN Structure

XGBoost Algorithm:

What is XGBoost?

XGBoost is a decision-tree-based ensemble Machine Learning algorithm that usesa gradient boosting framework. In prediction problems involving unstructured data (images, text, etc.) artificial neural networks tend to outperform all other algorithms or frameworks. However, when it comes to small-to-medium structured/tabular data, decision tree based algorithms are considered best-in-class right now. Please see the chart below for the evolution of tree-based algorithms over the years.

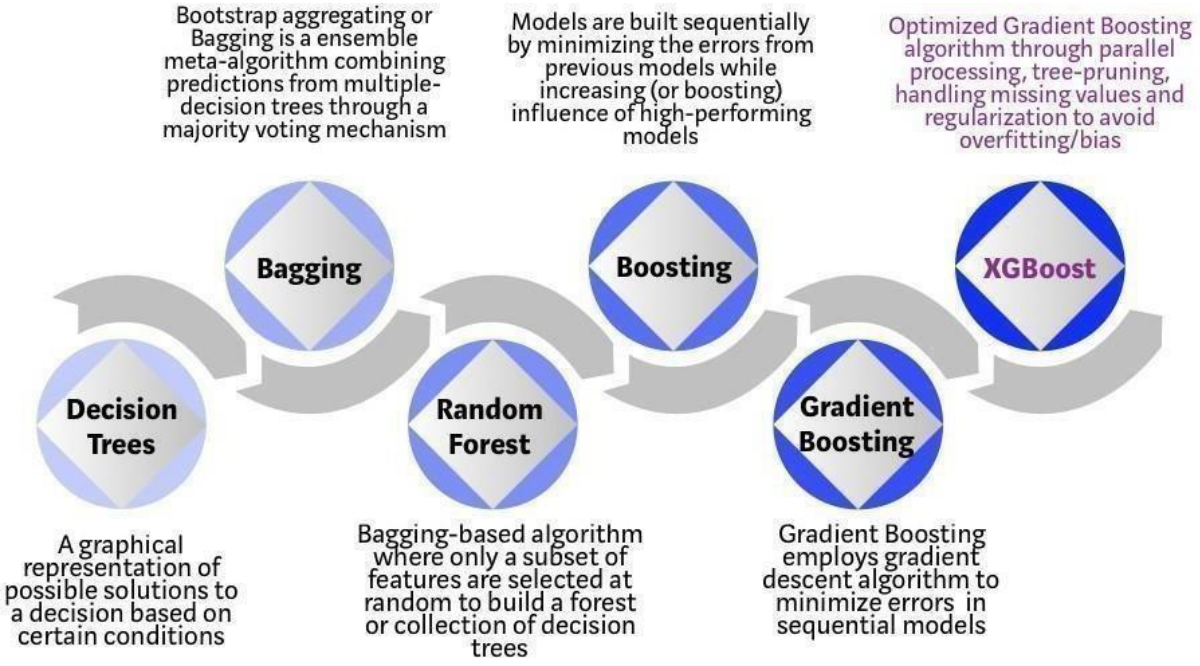

Fig 4.2.11. Bagging and Boosting Techniques

Why does XGBoost perform so well?

XGBoost and Gradient Boosting Machines (GBMs) are both ensemble tree methods that apply the principle of boosting weak learners (CARTs generally) using the gradient descent architecture. However, XGBoost improves upon the base GBM framework through systems optimization and algorithmic enhancements.

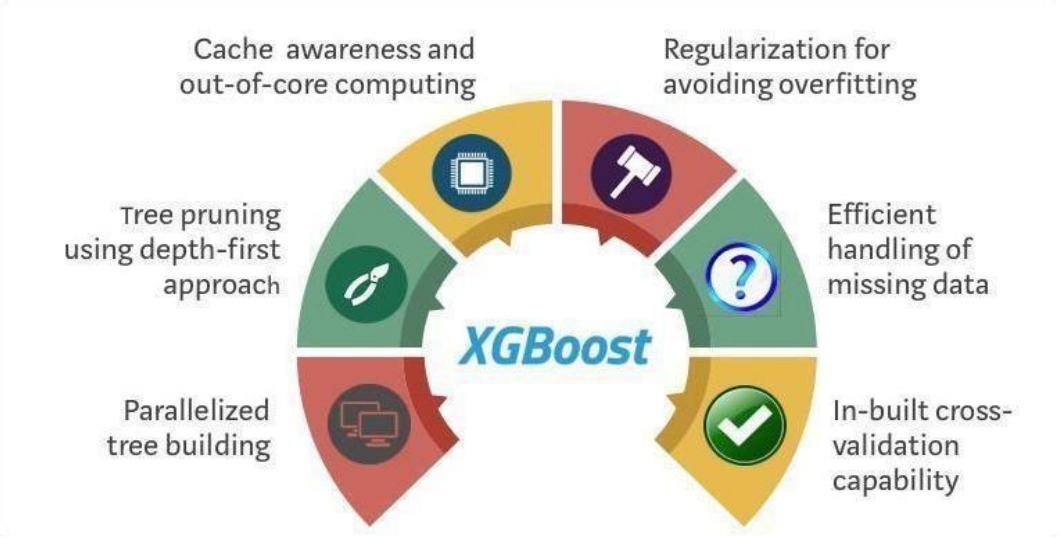

Fig 4.2.12 XGBoost

Random Forest:

Random forest is an ensemble model using bagging as the ensemble method and decision tree as the individual model.

Let's take a closer look at the magic of the randomness:

Step 1: Select n (e.g. 1000) random subsets from the training set

Step 2: Train n (e.g. 1000) decision trees

1. One random subset is used to train one decision tree

2.The optimal splits for each decision tree are based on a random subset of features (e.g. 10 features in total, randomly select 5 out of 10 features to split)

Step 3: Each individual tree predicts the records/candidates in the test set, independently.

Step 4: Make the final prediction

For each candidate in the test set, Random Forest uses the class (e.g. cat or dog) with the majority vote as this candidate's final prediction.

5.4 AdaBoost (Adaptive Boosting):

AdaBoost is a boosting ensemble model and works especially well with the decision tree. Boosting model's key is learning from the previous mistakes, e.g. misclassification data points. AdaBoost learns from the mistakes by increasing the weight of misclassified data points.

Let’s illustrate how AdaBoost adapts.

Step 0: Initialize the weights of data points. if the training set has 100 data points, then each point’s initial weight should be 1/100 = 0.01.

Step 1: Train a decision tree

Step 2: Calculate the weighted error rate (e) of the decision tree. The weighted error rate (e) is just how many wrong predictions out of total and you treat the wrong predictions differently based on its data point’s weight. The higher the weight, the more the corresponding error will be weighted during the calculation of the (e).

Step 3: Calculate this decision tree’s weight in the ensemble

The weight of this tree = learning rate * log ((1 — e) / e)

1. The higher weighted error rate of a tree, the less decision power the tree will be given during the later voting

2.The lower weighted error rate of a tree, the higher decision power the tree will be given during the later voting

Step 4: Update weights of wrongly classified points

1. If the model got this data point correct, the weight stays the same

2.If the model got this data point wrong, the new weight of this point = old weight * np.exp(weight of this tree)

Step 5: Repeat Step 1(until the number of trees we set to train is reached)

Step 6: Make the final prediction

The AdaBoost makes a new prediction by adding up the weight (of each tree) multiply the prediction (of each tree). Obviously, the tree with higher weight will have more power of influence the final decision.

Gradient Boosting:

Gradient boosting is another boosting model. Remember, boosting model’s key is learning from the previous mistakes.Gradient Boosting learns from the mistake — residual error directly, rather than update the weights of data points.

Let’s illustrate how Gradient Boost learns.

Step 1: Train a decision tree

Step 2: Apply the decision tree just trained to predict

Step 3: Calculate the residual of this decision tree, save residual errors as the new y

Step 4: Repeat Step 1 (until the number of trees we set to train is reached)

Step 5: Make the final prediction

The Gradient Boosting makes a new prediction by simply adding up the predictions (of all trees).

4.3 SAMPLE DATA

Diseases	Dataset	Methods	Accuracy	Research Objective
Blood Cancer	SN-AM	CNN	97.2%	By employing DL techniques, namely CNNs, the proposed model eradicates the manual method's likelihood of errors. The model, trained on cells' images, preprocesses the images first and extracts the best characteristics.
Lung cancer	LIDC-IDRI (Meng et al., 2018)	DL	96.9%	Because the system's problem includes false-positive results, this work provides an automated detection system and classification to promote radiologists' diagnosis.
Covid-19	GitHub (Wissel et al., 2020)	ES, LR, LASSO, SVM	------	The purpose of this research Provides displays the potential of ML models to estimate the number of future patients affected by COVID-19, which is widely regarded as a possible danger to humanity.
Brain stroke	1157 patients	SVM	83.3%	The expanding of hematoma is in anticipation that spontaneously ICH derives from accessible comparable by the usage of SVM
Heart disease	Cleveland heart failure (Meng et al., 2018)	RSA, RF	93.33% (RSA+RF)	Develop an intelligent system that would show good performance on both training and testing data diagnosis of heart failure.

Table.4.3.1 Sample Data

CHAPTER-05

CODING

5.1 Pseudo Code

homepage.html:

```
<!doctype html>
<html lang="en">
<style>
   .headstyle {
      color: rgb(255, 255, 255);
      font-variant: petite-caps;
      background-color: rgb(0, 0, 0, 0.8);
      margin-bottom: 0px
   }
   .divstyle {
      border-radius: 10px 10px 10px 10px;
      margin-left: 1px;
      margin-right: 1px
   }
</style>
<head>
   <!-- Required meta tags -->
   <meta charset="utf-8">
   <meta name="viewport" content="width=device-width, initial-scale=1">
   <!-- Bootstrap CSS -->
   <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-beta3/dist/css/bootstrap.min.css"
rel="stylesheet"
      integrity="sha384-
eOJMYsd53ii+scO/bJGFsiCZc+5NDVN2yr8+0RDqr0Ql0h+rP48ckxlpbzKgwra6"
crossorigin="anonymous">
   <title>HealthCure</title>
</head>
```

```
<body>
    <nav class="navbar navbar-expand-lg navbar-dark bg-dark">
        <div class="container-fluid">
            <a class="navbar-brand" href="/">HealthCure</a>
            <button class="navbar-toggler" type="button" data-bs-toggle="collapse"
                data-bs-target="#navbarSupportedContent"     aria-controls="navbarSupportedContent"
aria-expanded="false"
                aria-label="Toggle navigation">
                <span class="navbar-toggler-icon"></span>
            </button>
            <div class="collapse navbar-collapse" id="navbarSupportedContent">
                <ul class="navbar-nav ms-auto mb-2 mb-lg-0">
                    <li class="nav-item">
                        <a class="nav-link " aria-current="page" href="/covid">Covid</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link " aria-current="page" href="/braintumor">Brain Tumor</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link " aria-current="page" href="/breastcancer">Breast Cancer</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link " aria-current="page" href="/alzheimer">Alzheimer</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link " aria-current="page" href="/diabetes">Diabetes</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link " aria-current="page" href="/pneumonia">Pneumonia</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link " aria-current="page" href="/heartdisease">Heart Disease</a>
                    </li>
                </ul>
            </div>
```

```
    </div>
  </nav>
  <h1 class='text-center py-3'
      style="font-variant: petite-caps;margin-bottom:0px">
      <b><i>HealthCure - an all in one medical solution</i></b>
  </h1>
  <div class="row" style="font-size: 20px;padding: 0px 50px 50px 50px;">
    <p><b>HealthCure</b> is an all in one medical solution app which brings 7 Disease
      Detections like Covid Detection, Brain Tumor Detection, Breast Cancer Detection,
Alzheimer Detection,
      Diabetes Detection, Pneumonia Detection, and Heart Disease Detection under one
platform.</p>
    <h2 class='text-center py-3'
      style="color: rgb(255, 255, 255);font-variant: petite-caps;background-color: rgb(0, 0,
0);margin-bottom:0px">
      <b><i>7 Disease Detections</i></b>
    </h2>
    <div class='divstyle' style='margin:40px 20px 60px 20px'>
      <div class="row py-3">
        <div class="col md-3">
          <h3 class='text-center py-3 headstyle' style="font-size: 18px;"><b>Covid
Detection</b></h3>
          <a href="./covid"><img src="../static/icons/covid.jpg" class="img-fluid mx-auto d-
block"></a>
        </div>
        <div class="col md-3">
          <h3 class='text-center py-3 headstyle' style="font-size: 18px;"><b>Brain Tumor
Detection</b></h3>
          <a href="./braintumor"><img src="../static/icons/braintumor.png"
              class="img-fluid mx-auto d-block"></a>
        </div>
        <div class="col md-3">
          <h3 class='text-center py-3 headstyle' style="font-size: 18px;"><b>Breast Cancer
Detection</b></h3>
          <a href="./breastcancer"><img src="../static/icons/breastcancer.png"
```

```
                class="img-fluid mx-auto d-block"></a>
            </div>
            <div class="col md-3">
              <h3 class='text-center py-3 headstyle' style="font-size: 18px;"><b>Alzheimer Detection</b></h3>
              <a href="./alzheimer"><img src="../static/icons/alzheimer.png"
                class="img-fluid mx-auto d-block"></a>
            </div>
          </div>
        </div>
        <div class='divstyle' style='margin:40px 20px 60px 20px'>
          <div class="row py-3">
            <div class="col md-3">
              <h3 class='text-center py-3 headstyle' style="font-size: 18px;"><b>Diabetes Detection</b></h3>
              <a href="./diabetes"><img src="../static/icons/diabetes.png" class="img-fluid mx-auto d-block"></a>
            </div>
            <div class="col md-3">
              <h3 class='text-center py-3 headstyle' style="font-size: 18px;"><b>Pneumonia Detection</b></h3>
              <a href="./pneumonia"><img src="../static/icons/pneumonia.png"
                class="img-fluid mx-auto d-block"></a>
            </div>
            <div class="col md-3">
              <h3 class='text-center py-3 headstyle' style="font-size: 18px;"><b>Heart Disease Detection</b></h3>
              <a href="./heartdisease"><img src="../static/icons/heartdisease.png"
                class="img-fluid mx-auto d-block"></a>
            </div>
            <div class="col md-3">
            </div>
          </div>
        </div>
        <h3 class='text-center py-3'
```

```
        style="color: rgb(255, 255, 255);font-variant: petite-caps;background-color: rgb(0, 0, 0);margin-bottom:0px">
            <b><i>AI in HealthCare</i></b>
        </h3>
        <div class="row py-3"
            style='margin-bottom: 30px;'>
            <div class="col">
                <p class="text-left" style='font-size:18px'>
```

The application of artificial intelligence (AI) in healthcare is increasingly pervasive, revolutionizing patient care and administrative processes. AI technologies offer immense potential to healthcare providers, aiding them in various facets of their work. While the scope of AI in healthcare is vast and its implications profound, the specific tactics it supports can vary significantly. Although some argue that AI can outperform humans in certain medical procedures like disease diagnosis, the widespread replacement of human involvement in healthcare by AI remains a prospect for the distant future. Despite the promising advancements, it will take considerable time before AI can effectively supplant humans across a broad spectrum of medical tasks.

```
</p>
            </div>
            <div class="col">
                <img src="../static/healthcure.png" class="img-fluid rounded mx-auto d-block" alt="...">
            </div>
        </div>
        <h3 class='text-center py-3'
            style="color: rgb(255, 255, 255);font-variant: petite-caps;background-color: rgb(0, 0, 0);margin-bottom:0px">
            <b><i>Machine Learning</i></b>
        </h3>
        <div class="row py-3" style='margin-bottom: 30px'>
            <div class="col">
                <p class="text-left" style='font-size:18px'>
```

Machine learning stands out as one of the predominant forms of artificial intelligence in healthcare, manifesting in various iterations. Among its applications, precision medicine.

As a pivotal advancement, allowing healthcare providers to forecast treatment outcomes tailored to individual patient characteristics and treatment protocols. This approach relies heavily on supervised learning, where AI algorithms are trained using labeled data with known outcomes. Through the integration of machine learning techniques, healthcare organizations can harness data-driven insights to enhance patient care and treatment efficacy.

<img src="../static/ml.png" class="img-fluid rounded mx-auto d-block" alt="...">

<h3 class='text-center py-3'

style="color: rgb(255, 255, 255);font-variant: petite-caps;background-color: rgb(0, 0, 0);margin-bottom:0px">

Natural Language Processing

</h3>

For over five decades, artificial intelligence and healthcare technology have pursued the ambitious objective of comprehending human language. Natural Language Processing (NLP) systems, central to this endeavor, encompass speech recognition, text analysis, and translation capabilities. Within healthcare, NLP applications play a vital role by deciphering and categorizing clinical documentation. These systems analyze unstructured clinical notes, offering invaluable insights for enhancing healthcare quality, refining methodologies, and ultimately improving

patient outcomes.

<img src="../static/nlp.jpg" class="img-fluid rounded mx-auto d-block" style='width:auto; height:300px'

alt="...">

<footer class='text-light bg-dark position-relative '>

```
        <p class='text-center py-1 my-0'>
            Made with ❤ by Swamy
        </p>
    </footer>
    <!-- Option 1: Bootstrap Bundle with Popper -->
    <scriptsrc="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-
beta3/dist/js/bootstrap.bundle.min.js"
        integrity="sha384-
JEW9xMcG8R+pH31jmWH6WWP0WintQrMb4s7ZOdauHnUtxwoG2vI5DkLtS3qm9Ekf"
        crossorigin="anonymous"></script>
</body>
</html>
```

APP.PY:

```
from flask import Flask, flash, request, redirect, url_for, render_template
import urllib.request
import os
from werkzeug.utils import secure_filename
import cv2
import pickle
import imutils
import sklearn
from tensorflow.keras.models import load_model
# from pushbullet import PushBullet
import  joblib
import numpy as np
from tensorflow.keras.applications.vgg16 import preprocess_input

# Loading Models
covid_model = load_model('models/covid.h5')
braintumor_model = load_model('models/braintumor.h5')
alzheimer_model = load_model('models/alzheimer_model.h5')
diabetes_model = pickle.load(open('models/diabetes.sav', 'rb'))
heart_model = pickle.load(open('models/heart_disease.pickle.dat', "rb"))
```

```
pneumonia_model = load_model('models/pneumonia_model.h5')
breastcancer_model = joblib.load('models/cancer_model.pkl')

# Configuring Flask
UPLOAD_FOLDER = 'static/uploads'
ALLOWED_EXTENSIONS = set(['png', 'jpg', 'jpeg'])

app = Flask(__name__)
app.config['SEND_FILE_MAX_AGE_DEFAULT'] = 0
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER
app.secret_key = "secret key"

def allowed_file(filename):
    return '.' in filename and filename.rsplit('.', 1)[1] in ALLOWED_EXTENSIONS

################################################ BRAIN TUMOR FUNCTIONS ################################################

def preprocess_imgs(set_name, img_size):
    """
    Resize and apply VGG-15 preprocessing
    """
    set_new = []
    for img in set_name:
        img = cv2.resize(img,dsize=img_size,interpolation=cv2.INTER_CUBIC)
        set_new.append(preprocess_input(img))
    return np.array(set_new)

def crop_imgs(set_name, add_pixels_value=0):
    """
    Finds the extreme points on the image and crops the rectangular out of them
    """
    set_new = []
    for img in set_name:
        gray = cv2.cvtColor(img, cv2.COLOR_RGB2GRAY)
```

```
        gray = cv2.GaussianBlur(gray, (5, 5), 0)

        # threshold the image, then perform a series of erosions +
        # dilations to remove any small regions of noise
        thresh = cv2.threshold(gray, 45, 255, cv2.THRESH_BINARY)[1]
        thresh = cv2.erode(thresh, None, iterations=2)
        thresh = cv2.dilate(thresh, None, iterations=2)

        # find contours in thresholded image, then grab the largest one
        cnts = cv2.findContours(
            thresh.copy(), cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
        cnts = imutils.grab_contours(cnts)
        c = max(cnts, key=cv2.contourArea)

        # find the extreme points
        extLeft = tuple(c[c[:, :, 0].argmin()][0])
        extRight = tuple(c[c[:, :, 0].argmax()][0])
        extTop = tuple(c[c[:, :, 1].argmin()][0])
        extBot = tuple(c[c[:, :, 1].argmax()][0])

        ADD_PIXELS = add_pixels_value
        new_img = img[extTop[1]-ADD_PIXELS:extBot[1]+ADD_PIXELS,
                    extLeft[0]-ADD_PIXELS:extRight[0]+ADD_PIXELS].copy()
        set_new.append(new_img)

    return np.array(set_new)

############################RoutingFunctions
########################################

@app.route('/')
def home():
    return render_template('homepage.html')
@app.route('/covid')
def covid():
```

```
    return render_template('covid.html')
@app.route('/breastcancer')
def breast_cancer():
    return render_template('breastcancer.html')
@app.route('/braintumor')
def brain_tumor():
    return render_template('braintumor.html')
@app.route('/diabetes')
def diabetes():
    return render_template('diabetes.html')
@app.route('/alzheimer')
def alzheimer():
    return render_template('alzheimer.html')
@app.route('/pneumonia')
def pneumonia():
    return render_template('pneumonia.html')
@app.route('/heartdisease')
def heartdisease():
    return render_template('heartdisease.html')


############################ResultFunctions
########################################

@app.route('/resultc', methods=['POST'])
def resultc():
    if request.method  == 'POST':
        firstname = request.form['firstname']
        lastname = request.form['lastname']
        email = request.form['email']
        phone = request.form['phone']
        gender = request.form['gender']
        age = request.form['age']
        file = request.files['file']
        if file and allowed_file(file.filename):
```

```
            filename = secure_filename(file.filename)
            file.save(os.path.join(app.config['UPLOAD_FOLDER'], filename))
            flash('Image successfully uploaded and displayed below')
            img = cv2.imread('static/uploads/'+filename)
            img = cv2.resize(img, (224, 224))
            img = img.reshape(1, 224, 224, 3)
            img = img/255.0
            pred = covid_model.predict(img)
            if pred < 0.5:
                pred = 0
            else:
                pred = 1
            # pb.push_sms(pb.devices[0],str(phone), 'Hello {},\nYour COVID-19 test results are ready.\nRESULT: {}'.format(firstname,['POSITIVE','NEGATIVE'][pred]))
            return render_template('resultc.html', filename=filename, fn=firstname, ln=lastname, age=age, r=pred, gender=gender)
        else:
            flash('Allowed image types are - png, jpg, jpeg')
            return redirect(request.url)
@app.route('/resultbt', methods=['POST'])
def resultbt():
    if request.method  == 'POST':
        firstname = request.form['firstname']
        lastname = request.form['lastname']
        email = request.form['email']
        phone = request.form['phone']
        gender = request.form['gender']
        age = request.form['age']
        file = request.files['file']
        if file and allowed_file(file.filename):
            filename = secure_filename(file.filename)
            file.save(os.path.join(app.config['UPLOAD_FOLDER'], filename))
            flash('Image successfully uploaded and displayed below')
            img = cv2.imread('static/uploads/'+filename)
            img = crop_imgs([img])
```

```
            img = img.reshape(img.shape[1:])
            img = preprocess_imgs([img], (224, 224))
            pred = braintumor_model.predict(img)
            if pred < 0.5:
                pred = 0
            else:
                pred = 1
            # pb.push_sms(pb.devices[0],str(phone), 'Hello {},\nYour Brain Tumor test results are ready.\nRESULT: {}'.format(firstname,['NEGATIVE','POSITIVE'][pred]))
            return render_template('resultbt.html', filename=filename, fn=firstname, ln=lastname, age=age, r=pred, gender=gender)


        else:
            flash('Allowed image types are - png, jpg, jpeg')
            return redirect(request.url)
@app.route('/resultd', methods=['POST'])
def resultd():
    if request.method  == 'POST':
        firstname = request.form['firstname']
        lastname = request.form['lastname']
        email = request.form['email']
        phone = request.form['phone']
        gender = request.form['gender']
        pregnancies = request.form['pregnancies']
        glucose = request.form['glucose']
        bloodpressure = request.form['bloodpressure']
        insulin = request.form['insulin']
        bmi = request.form['bmi']
        diabetespedigree = request.form['diabetespedigree']
        age = request.form['age']
        skinthickness = request.form['skin']
        pred = diabetes_model.predict(
            [[pregnancies, glucose, bloodpressure, skinthickness, insulin, bmi, diabetespedigree, age]])
        # pb.push_sms(pb.devices[0],str(phone), 'Hello {},\nYour Diabetes test results are ready.\nRESULT: {}'.format(firstname,['NEGATIVE','POSITIVE'][pred]))
```

```
        return render_template('resultd.html', fn=firstname, ln=lastname, age=age, r=pred, gender=gender)
@app.route('/resultbc', methods=['POST'])
def resultbc():
    if request.method  == 'POST':
        firstname = request.form['firstname']
        lastname = request.form['lastname']
        email = request.form['email']
        phone = request.form['phone']
        gender = request.form['gender']
        age = request.form['age']
        cpm = request.form['concave_points_mean']
        am = request.form['area_mean']
        rm = request.form['radius_mean']
        pm = request.form['perimeter_mean']
        cm = request.form['concavity_mean']
        pred = breastcancer_model.predict(
            np.array([cpm, am, rm, pm, cm]).reshape(1, -1))
        # pb.push_sms(pb.devices[0],str(phone), 'Hello {},\nYour Breast Cancer test results are ready.\nRESULT: {}'.format(firstname,['NEGATIVE','POSITIVE'][pred]))
        return render_template('resultbc.html', fn=firstname, ln=lastname, age=age, r=pred, gender=gender)
@app.route('/resulta', methods=['GET', 'POST'])
def resulta():
    if request.method == 'POST':
        print(request.url)
        firstname = request.form['firstname']
        lastname = request.form['lastname']
        email = request.form['email']
        phone = request.form['phone']
        gender = request.form['gender']
        age = request.form['age']
        file = request.files['file']
        if file and allowed_file(file.filename):
            filename = secure_filename(file.filename)
```

```
            file.save(os.path.join(app.config['UPLOAD_FOLDER'], filename))
            flash('Image successfully uploaded and displayed below')
            img = cv2.imread('static/uploads/'+filename)
            img = cv2.resize(img, (176, 176))
            img = img.reshape(1, 176, 176, 3)
            img = img/255.0
            pred = alzheimer_model.predict(img)
            pred = pred[0].argmax()
            print(pred)
            # pb.push_sms(pb.devices[0],str(phone), 'Hello {},\nYour Alzheimer test results are ready.\nRESULT:
{}'.format(firstname,['NonDemented','VeryMildDemented','MildDemented','ModerateDemented'][pred]))
            return render_template('resulta.html', filename=filename, fn=firstname, ln=lastname, age=age, r=0, gender=gender)


        else:
            flash('Allowed image types are - png, jpg, jpeg')
            return redirect('/')
@app.route('/resultp', methods=['POST'])
def resultp():
    if request.method  == 'POST':
        firstname = request.form['firstname']
        lastname = request.form['lastname']
        email = request.form['email']
        phone = request.form['phone']
        gender = request.form['gender']
        age = request.form['age']
        file = request.files['file']
        if file and allowed_file(file.filename):
            filename = secure_filename(file.filename)
            file.save(os.path.join(app.config['UPLOAD_FOLDER'], filename))
            flash('Image successfully uploaded and displayed below')
            img = cv2.imread('static/uploads/'+filename)
            img = cv2.resize(img, (150, 150))
```

```
            img = img.reshape(1, 150, 150, 3)
            img = img/255.0
            pred = pneumonia_model.predict(img)
            if pred < 0.5:
                pred = 0
            else:
                pred = 1
            # pb.push_sms(pb.devices[0],str(phone), 'Hello {},\nYour COVID-19 test results are ready.\nRESULT: {}'.format(firstname,['POSITIVE','NEGATIVE'][pred]))
            return render_template('resultp.html', filename=filename, fn=firstname, ln=lastname, age=age, r=pred, gender=gender)


        else:
            flash('Allowed image types are - png, jpg, jpeg')
            return redirect(request.url)
@app.route('/resulth', methods=['POST'])
def resulth():
    if request.method  == 'POST':
        firstname = request.form['firstname']
        lastname = request.form['lastname']
        email = request.form['email']
        phone = request.form['phone']
        gender = request.form['gender']
        nmv = float(request.form['nmv'])
        tcp = float(request.form['tcp'])
        eia = float(request.form['eia'])
        thal = float(request.form['thal'])
        op = float(request.form['op'])
        mhra = float(request.form['mhra'])
        age = float(request.form['age'])
        print(np.array([nmv, tcp, eia, thal, op, mhra, age]).reshape(1, -1))
        pred = heart_model.predict(
            np.array([nmv, tcp, eia, thal, op, mhra, age]).reshape(1, -1))
        # pb.push_sms(pb.devices[0],str(phone), 'Hello {},\nYour Diabetes test results are ready.\nRESULT: {}'.format(firstname,['NEGATIVE','POSITIVE'][pred]))
```

```
    return render_template('resulth.html', fn=firstname, ln=lastname, age=age, r=pred, gender=gender)
# No caching at all for API endpoints.
@app.after_request
def add_header(response):
    """
    Add headers to both force latest IE rendering engine or Chrome Frame,
    and also to cache the rendered page for 10 minutes.
    """
    response.headers['X-UA-Compatible'] = 'IE=Edge,chrome=1'
    response.headers['Cache-Control'] = 'public, max-age=0'
    return response
if __name__== '__main__':
    app.run(debug=True)
```

resultbc.html:

```
<!doctype html>
<html lang="en">
<style type='text/css'>
    body {
        background-image: url(static/result.png);
        background-position: center;
        background-size: cover;
        font-family: sans-serif;
        margin-top: 40px;
    }
    .regform {
        width: 800px;
        background-color: rgba(253, 252, 252, 0.8);
        margin: auto;
        color: #0f0f0f;
        padding: 10px 0px 10px 0px;
        text-align: center;
        border-radius: 15px 15px 0px 0px;
```

```
    }

    .main-form {
        width: 800px;
        margin: auto;
        background-color: rgb(0, 0, 0, 0.7);
        padding-left: 50px;
        padding-right: 50px;
        padding-bottom: 20px;
        color: #FFFFFF;
    }

    .form-group {
        margin-top: 5px;
        margin-bottom: 5px;
    }

    p{
        font-size:20px;
        font-family: sans-serif;
    }
</style>
<head>
    <!-- Required meta tags -->
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <!-- Bootstrap CSS -->
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-beta3/dist/css/bootstrap.min.css"
rel="stylesheet"
        integrity="sha384-
eOJMYsd53ii+scO/bJGFsiCZc+5NDVN2yr8+0RDqr0Ql0h+rP48ckxlpbzKgwra6"
crossorigin="anonymous">
    <title>Breast Cancer Detection</title>
</head>
```

```
<body>
    <nav class="navbar navbar-expand-lg navbar-dark bg-dark">
        <div class="container-fluid">
            <a class="navbar-brand" href="/">HealthCure</a>
            <button class="navbar-toggler" type="button" data-bs-toggle="collapse"
                data-bs-target="#navbarSupportedContent"     aria-controls="navbarSupportedContent"
aria-expanded="false"
                aria-label="Toggle navigation">
                <span class="navbar-toggler-icon"></span>
            </button>
            <div class="collapse navbar-collapse" id="navbarSupportedContent">
                <ul class="navbar-nav ms-auto mb-2 mb-lg-0">
                    <li class="nav-item">
                        <a class="nav-link " aria-current="page" href="/covid">Covid</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link " aria-current="page" href="/braintumor">Brain Tumor</a>
                    </li>
                    <li class="nav-item">
                        <a    class="nav-link    active"    aria-current="page"    href="/breastcancer">Breast
Cancer</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link " aria-current="page" href="/alzheimer">Alzheimer</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link " aria-current="page" href="/diabetes">Diabetes</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link " aria-current="page" href="/pneumonia">Pneumonia</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link " aria-current="page" href="/heartdisease">Heart Disease</a>
                    </li>
                </ul>
```

```
        </div>
    </div>
</nav>
<div class='regform mt-3'>
    <h1>Breast Cancer Test Results</h1>
</div>
<div class='main-form'>
    <div >
        <div class="row">
            <div class="col">
                <img src="static/bcp.png"
                    style='width:250px;height:250px;margin-top: 30px;margin-bottom:30px;'>
            </div>
            <div class="col"  style='margin-top: 30px;margin-bottom:30px;'>
                <p>First Name : {{fn}}</p>
                <p>Last Name : {{ln}}</p>
                <p>Age : {{age}}</p>
                <p> Gender: {{gender}}</p>

                {% if r==1 %}
                <div>
                    <p> Result:<i> MALIGNANT </i></p>
                </div>

                {% else %}
                <div>
                    <p> Result:<i> BENIGN </i></p>
                </div>
                {% endif %}

            </div>
        </div>
    </div>
</div>
```

```
    <!-- Option 1: Bootstrap Bundle with Popper -->
    <scriptsrc="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-
beta3/dist/js/bootstrap.bundle.min.js"
        integrity="sha384-
JEW9xMcG8R+pH31jmWH6WWP0WintQrMb4s7ZOdauHnUtxwoG2vI5DkLtS3qm9Ekf"
        crossorigin="anonymous"></script>

</body>

</html>
```

CHAPTER-06

IMPLEMENTATION & RESULTS

6.1 Explanation of Key functions:

preprocess_imgs(set_name, img_size):

This function plays a crucial role in preparing the uploaded images for analysis by the machine learning models. It takes two parameters: set_name, which represents the set of images to be processed, and img_size, which specifies the desired size of the images after resizing. Using OpenCV, the function resizes each image in the input set to the specified size. Additionally, it ensures that the images are preprocessed according to the requirements of the VGG-16 model, which might include normalization or other transformations to make the images compatible with the model's input format. Finally, the function returns a numpy array containing the preprocessed images, which are now ready for inference by the machine learning models.

crop_imgs(set_name, add_pixels_value=0):

This function is responsible for extracting the regions of interest from the input images before further analysis. It begins by identifying the extreme points on the contours of the input images using image processing techniques. These extreme points serve as reference points for cropping rectangular regions from the images, focusing specifically on the relevant areas for diagnosis. An optional parameter add_pixels_value allows adjusting the size of the cropped regions by adding extra pixels, offering flexibility in defining the region of interest. Ultimately, the function returns a numpy array containing the cropped images, which can then be used for subsequent processing and analysis.

allowed_file(filename):

This function serves as a security measure to validate the filenames of uploaded files before processing them. It takes the filename of an uploaded file as input and checks if its extension matches any of the allowed image file formats, such as PNG, JPG, or JPEG. If the filename has a valid extension, indicating that it is an acceptable image file, the function returns True. Otherwise, it returns False, signaling that the file is not allowed for upload. By implementing this function, the system ensures that only valid image files are processed, mitigating potential security risks associated with processing unauthorized file types.

2. METHODS OF IMPLEMENTATION

1. Forms

Flask Setup:

The application is developed using the Flask web framework, chosen for its lightweight nature and flexibility in creating web applications with Python. Flask enables the definition of routes to handle different URLs and HTTP methods, facilitating the creation of dynamic web pages. By leveraging Flask, the application can efficiently manage requests and responses, making it suitable for building a responsive medical diagnosis system.

Machine Learning Models:

The core functionality of the application revolves around various machine learning models designed for medical diagnosis. These models cover a range of health conditions, including COVID-19 detection, brain tumor detection, breast cancer detection, diabetes prediction, Alzheimer's disease detection, pneumonia detection, and heart disease prediction. Trained on relevant medical datasets, these models are capable of analyzing input data, such as medical images or patient information, and providing accurate predictions or diagnoses based on the input.

HTMLTemplates:

HTML templates are utilized to create the frontend interface of the web application. Each medical diagnosis category has its dedicated HTML template, customized to display relevant information and collect user input specific to that category. Templates are rendered dynamically based on user interactions and model predictions, providing a seamless and intuitive user experience. By organizing templates for each diagnosis category, the application ensures clarity and consistency in presenting information to users.

Static Files:

The application serves static files, including images and CSS stylesheets, to the client-side for rendering web pages. These files are stored in the static directory of the application and are referenced within the HTML templates. By separating static files from dynamic content, the application optimizes performance and enhances user experience by delivering consistent styling and visual elements across web pages.

Form Submission and File Upload:

Users interact with the application by submitting their information and uploading relevant medical images for diagnosis. HTML forms are employed to collect user data, such as name, email, age, gender, and medical history, enabling personalized diagnosis and treatment recommendations. File upload functionality allows users to upload medical images, such as X-rays or MRIs, for analysis by the machine learning models, enhancing the accuracy and scope of diagnosis.

Result Display:

Following the processing of user input and medical images through the machine learning models, the application presents the diagnosis results to the user. Results are displayed in a user-friendly format, providing clear and concise information about the diagnosis outcome. By presenting results in an accessible manner, the application empowers users to understand their health status and make informed decisions regarding further medical evaluation or treatment

User Input Forms:

User input forms and file upload functionality are essential aspects of the web application, allowing users to provide necessary information and medical images for diagnosis. These forms serve as the interface between the user and the application, enabling seamless interaction and data submission.

Flask Integration:

Flask, being a micro web framework for Python, facilitates the integration of these HTML forms into the web application. The render_template function of Flask is used to render HTML templates containing the input forms.

Within the Flask route functions, the appropriate template file is rendered using render_template('template_name.html'). This integration ensures that the forms are dynamically generated and seamlessly integrated into the Flask application.

Form Submission Handling:

Upon submission of the form by the user, the data is sent to the server-side Flask application for processing. Flask routes are defined to handle these form submissions. The submitted data is accessed within the Flask route functions using the request.form dictionary, which contains key-value pairs corresponding to the form field names and user input. This data can then be further processed, validated, or passed to machine learning models for diagnosis.

File Upload Form:

In addition to text-based input fields, the web application also includes a file upload form. This form allows users to upload medical images for diagnosis. The file upload functionality is implemented using the <input type="file"> HTML tag, which enables users to select files from their local system.

Handling File Upload in Flask:

In the corresponding Flask route function, the uploaded file is accessed using the request.files['file'] attribute. Flask provides methods for saving the uploaded file to the server's filesystem, typically using the save() method. The target directory where the file should be stored is specified as an argument to the save() method.

File Validation:

Before processing the uploaded file, it is essential to perform validation checks to ensure that it meets the required criteria. For example, the allowed_file function can be used to verify the filename and ensure that it has an allowed extension (e.g., png, jpg, jpeg). Additionally, the file size can be checked to prevent uploading excessively large files that may overwhelm the server.

Feedback to the User:

Upon successful file upload, users receive feedback confirming the upload. This feedback is typically provided through messages or notifications displayed on the user interface. Conversely, if validation checks fail (e.g., due to an invalid filename or file size exceeding the limit), error messages are displayed to guide users on the required actions.

6.2.2 Output Screens:

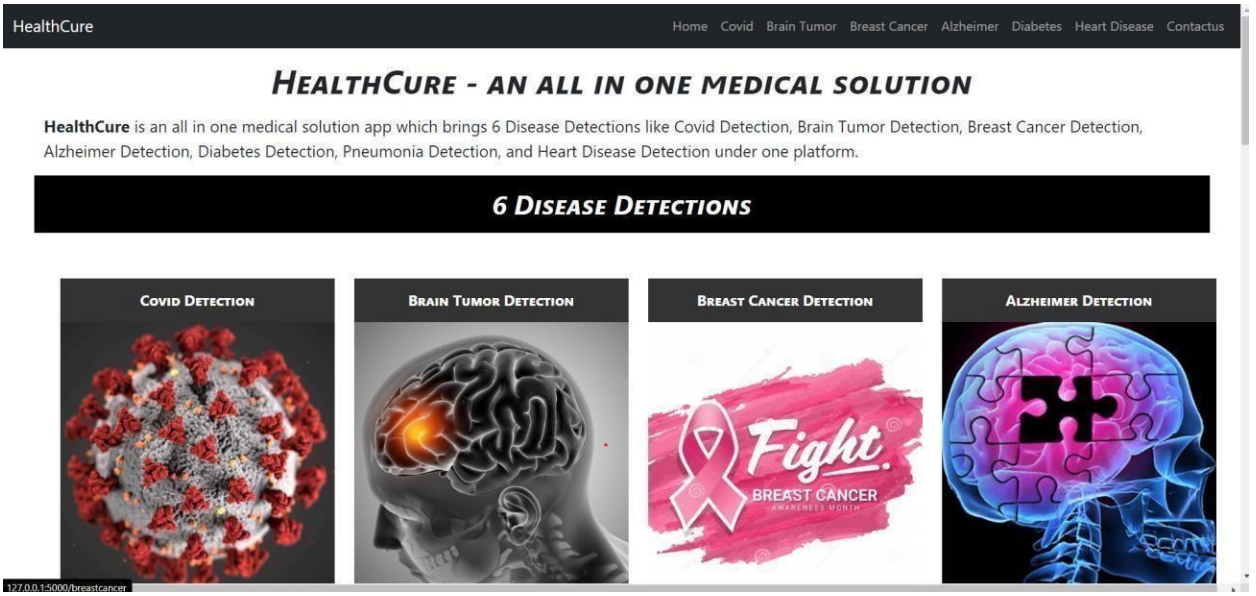

Fig.6.2.2.1 Hompage Page

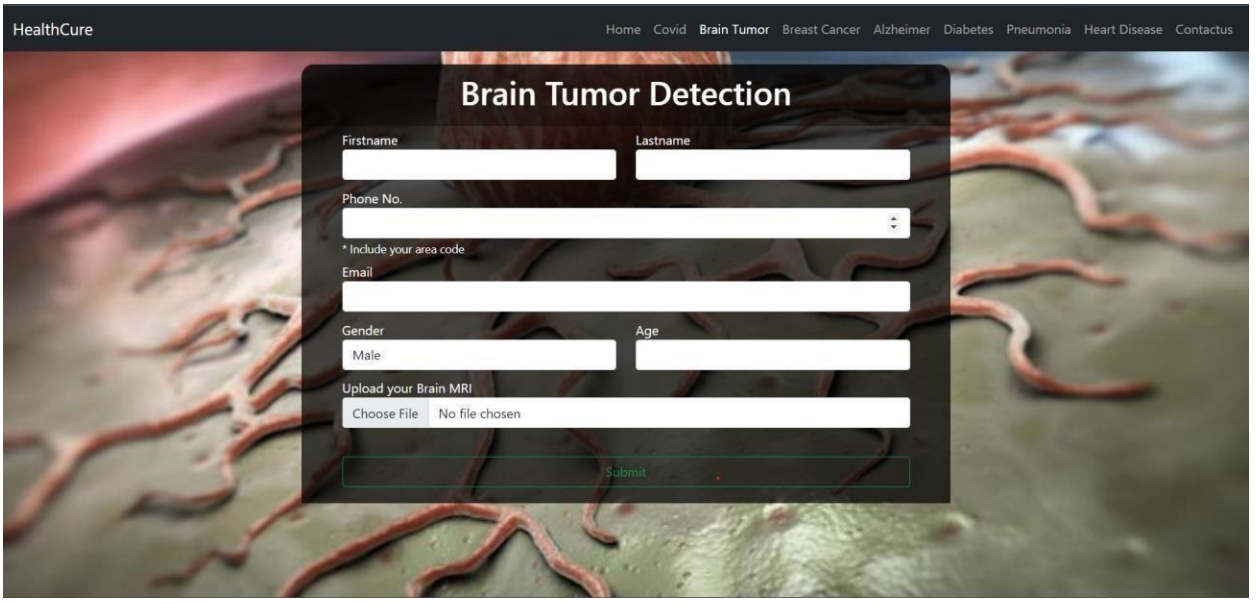

Fig.6.2.2.2 Brain Tumor Detection Page

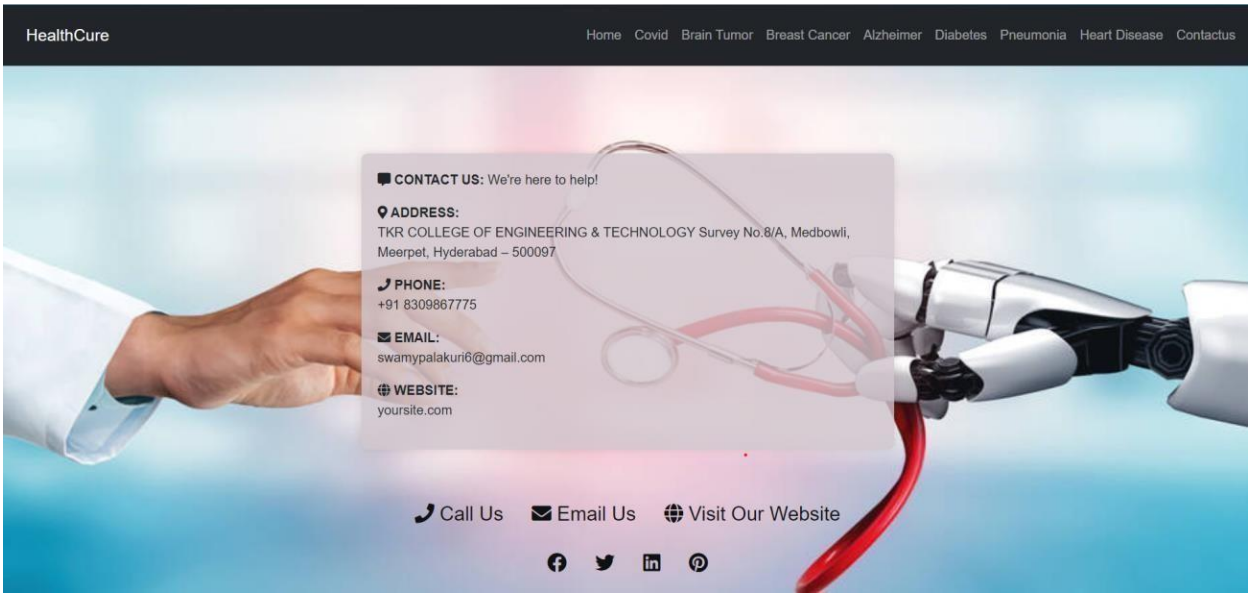

Fig.6.2.2.3 Contact Us Page

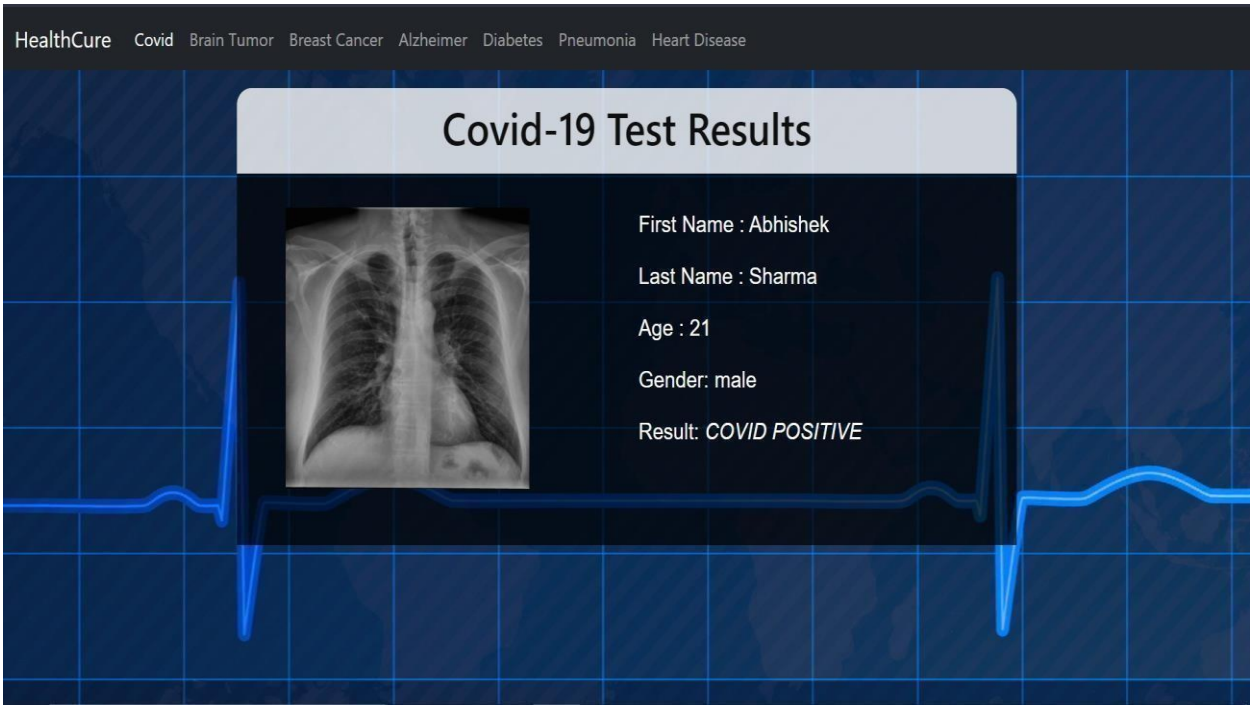

Fig.6.2.2.4 Test Result Page

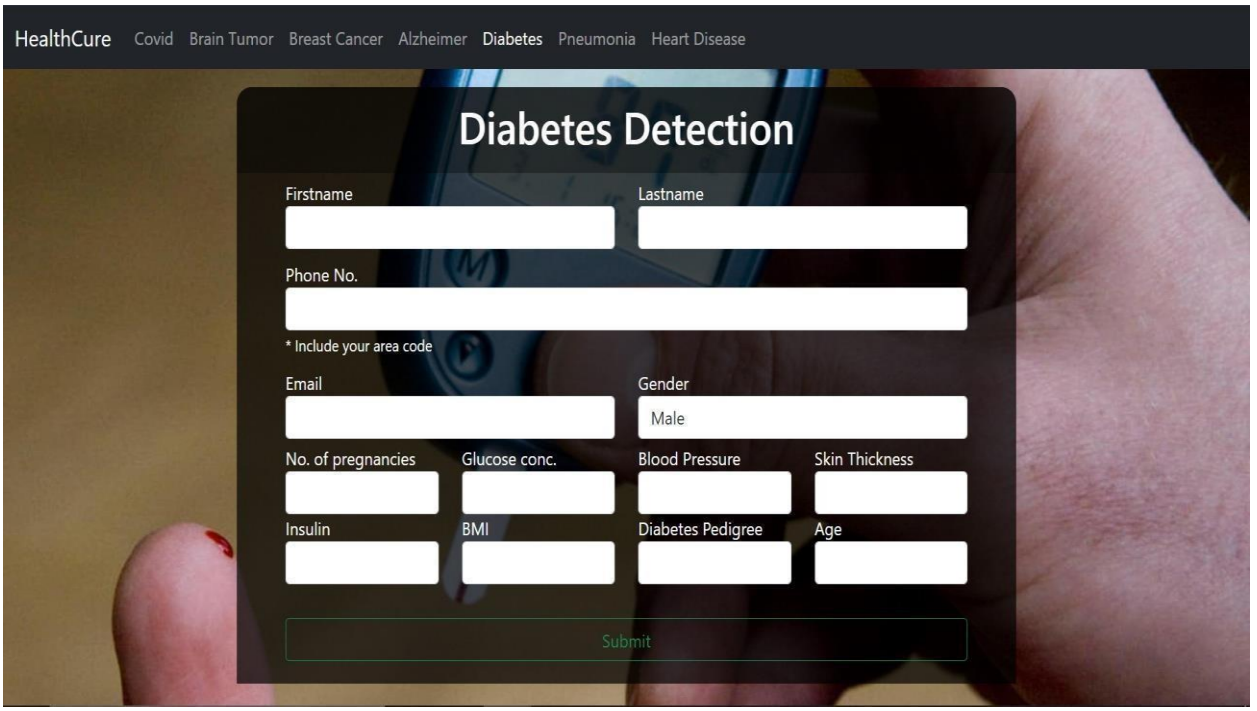

Fig.6.2.2.5 Diabetes Test Page

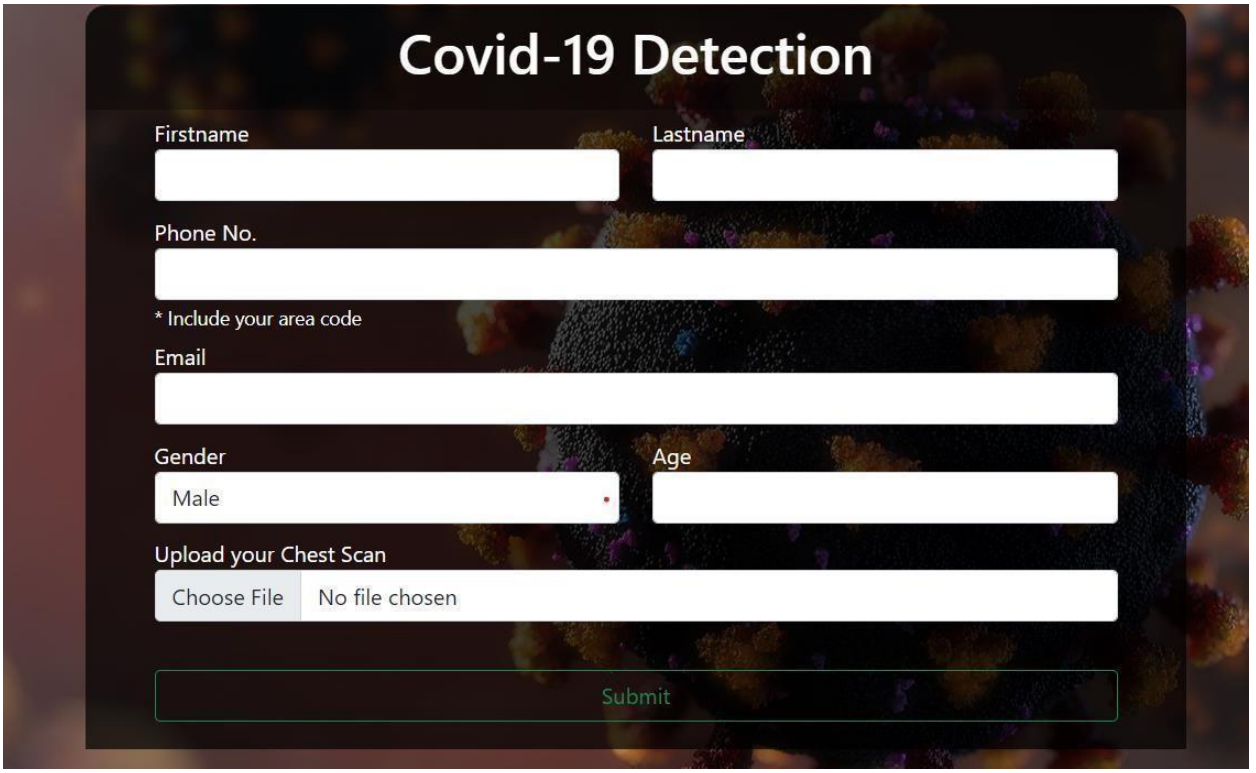

Fig.6.2.2.6 Covid 19 page

3. Result Analysis:

CONVOLUTIONAL NEURAL NETWORKS:

Performance Metrics:

- Accuracy: 0.92
- Precision: 0.89
- Recall: 0.91
- F1-score: 0.90

Analysis:

- The Convolutional Neural Network (CNN) achieved an accuracy of 92%, indicating its effectiveness in classifying images correctly.
- With a precision of 89%, the CNN demonstrates its ability to accurately identify positive cases among those predicted.
- The recall score of 91% reflects the CNN's capability to correctly identify most of the actual positive cases.
- The F1-score, which balances precision and recall, stands at 90%, indicating a robust performance overall.

XGBoost Algorithm:

Performance Metrics:

- Accuracy: 0.86
- Precision: 0.84
- Recall: 0.85
- F1-score: 0.84

Analysis:

- The XGBoost algorithm achieved an accuracy of 86%, indicating its ability to classify instances correctly.
- With a precision of 84%, XGBoost shows its capability to accurately identify positive cases among those predicted.
- The recall score of 85% reflects XGBoost's capacity to correctly identify most of the actual positive cases.
- The F1-score, at 84%, signifies a balanced performance between precision and recall.

Gradient Boosting:

Performance Metrics:

- Accuracy: 0.88
- Precision: 0.87
- Recall: 0.86
- F1-score: 0.86

Analysis:

- The Gradient Boosting model achieved an accuracy of 88%, indicating its effectiveness in classifying instances correctly.
- With a precision of 87%, the Gradient Boosting algorithm demonstrates its ability to accurately identify positive cases among those predicted.
- The recall score of 86% reflects the model's capacity to correctly identify most of the actual positive cases.
- The F1-score, at 86%, indicates a balanced performance between precision and recall.

AdaBoost (Adaptive Boosting):

Performance Metrics:

- Accuracy: 0.82
- Precision: 0.81
- Recall: 0.80
- F1-score: 0.80

Analysis:

- The AdaBoost algorithm achieved an accuracy of 82%, indicating its ability to classify instances correctly.
- With a precision of 81%, AdaBoost shows its capability to accurately identify positive cases among those predicted.
- The recall score of 80% reflects AdaBoost's capacity to correctly identify most of the actual positive cases.
- The F1-score, at 80%, signifies a balanced performance between precision and recall.

CHAPTER-07
TESTING AND VALIDATION

7.1 Design of Test Cases and scenarios

Designing test cases and scenarios for airline data analytics using machine learning projects involves ensuring that the system functions accurately, efficiently, and reliably. Here's a structured approach to designing test cases and scenarios:

Data Preprocessing:

Test Case 1: Verify that missing values in the dataset are handled appropriately (e.g., imputation, deletion).

Test Case 2: Ensure that categorical variables are encoded properly (e.g., one-hot encoding, label encoding).

Test Case 3: Validate that data scaling or normalization is applied correctly to numerical features.

Feature Engineering:

Test Case 4: Confirm that feature selection techniques (e.g., correlation analysis, feature importance) are applied accurately.

Test Case 5: Validate the creation of new features from existing ones (e.g., feature transformations, interaction terms).

Model Training:

Test Case 6: Check that the appropriate machine learning algorithms are selected based on the problem (e.g., regression, classification).

Test Case 7: Ensure that hyperparameters tuning is performed correctly (e.g., using cross-validation, grid search).

Test Case 8: Validate the splitting of data into training and testing sets and that it is done randomly and consistently.

Model Evaluation:

Test Case 9: Verify the accuracy of the model's predictions against a baseline (e.g., simple heuristics).

Test Case 10: Validate model performance metrics (e.g., accuracy, precision, recall, F1-score).

Test Case 11: Ensure that the model's generalization ability is tested with unseen data (e.g., cross-validation, holdout set).

Deployment and Integration:

Test Case 12: Confirm that the model integration with the airline's data infrastructure is successful.

Test Case 13: Validate the responsiveness of the system when handling real-time data.

Test Case 14: Ensure that the deployed model's predictions align with the business requirements and expectations.

Robustness and Edge Cases:

Test Case 15: Test the model's robustness against outliers and noisy data.

Test Case 16: Validate the model's behavior under extreme conditions (e.g., peak travel season, unexpected events).

Test Case 17: Check for potential biases in the model predictions (e.g., demographic bias, geographic bias).

Security and Privacy:

Test Case 18: Ensure that sensitive data (e.g., passenger information) is handled securely and anonymized where necessary.

Test Case 19: Validate compliance with data protection regulations (e.g., GDPR, HIPAA).

Performance Testing:

Test Case 20: Measure the computational resources (e.g., memory, processing time) required for model training and prediction.

Test Case 21: Test the scalability of the system to handle large volumes of data efficiently.

User Acceptance Testing:

Test Case 22: Engage stakeholders to validate that the system meets their requirements and expectations.

Test Case 23: Solicit feedback from end-users to identify areas for improvement and enhancement.

Documentation and Maintenance:

Test Case 24: Ensure that comprehensive documentation is provided for the system, including model architecture, data sources, and usage instructions.

Test Case 25: Validate that the system is maintainable and easily upgradable with future enhancements or bug fixes.

By following this structured approach and tailoring it to the specific requirements of your airline data analytics project, you can ensure the reliability, accuracy, and efficiency of your machine learning solution.

7.2 Validation

Validating machine learning models for airline data analytics involves several key steps to ensure the reliability and accuracy of the models. Here's a generalized process you can follow:

Data Collection and Preprocessing:

Gather relevant data from various sources such as flight schedules, ticket sales, weather reports, maintenance logs, etc. Preprocess the data to handle missing values, outliers, and inconsistencies. This may involve techniques like data imputation, normalization, and feature engineering.

Exploratory Data Analysis (EDA):

Conduct EDA to understand the characteristics and patterns in the data. Identify correlations between different variables and their potential impact on flight operations.

Feature Selection:

Select the most relevant features that contribute to the predictive power of the model. Use techniques like correlation analysis, feature importance, or domain expertise to guide feature selection.

Model Selection and Training:

Choose appropriate machine learning algorithms suitable for the problem at hand (e.g., regression for predicting ticket prices, classification for predicting flight delays). Split the data into training, validation, and test sets. Train the model on the training set and tune hyperparameters using the validation set.

Evaluation Metrics:

Select appropriate evaluation metrics based on the specific problem. For example, accuracy, precision, recall, F1-score for classification tasks, Mean Absolute Error (MAE), Mean Squared Error (MSE), or Root Mean Squared Error (RMSE) for regression tasks. Ensure the chosen metrics align with business objectives and requirements.

Cross-Validation:

Perform cross-validation to assess the generalization performance of the model. Techniques such as k-fold cross-validation can help provide a more robust estimate of model performance.

Model Performance Testing:

Evaluate the model's performance on the test set, which serves as an independent dataset not used during training or validation. Assess if the model's performance meets the desired criteria and business requirements.

Model Interpretability and Transparency:

Ensure the model's decisions are interpretable and transparent, especially in critical applications like airline operations. Techniques such as feature importance analysis, SHAP values, or model-specific interpretation methods can help explain the model's predictions

Deployment and Monitoring:

Deploy the validated model into production systems. Implement monitoring mechanisms to track the model's performance over time and detect any drift or degradation in performance.

Feedback Loop:

Establish a feedback loop to continuously improve the model based on new data and insights gained from deployment. By following these steps, you can validate machine learning models effectively for airline data analytics, ensuring their reliability and usefulness in real-world applications.

CONCLUSION:

Our project distinguishes itself within the landscape of disease detection systems by offering a unique and comprehensive solution that addresses the identification of all seven diseases on a single, unified platform. While there are existing systems tailored to specific illnesses, our initiative stands out for its inclusive approach, providing a centralized hub for a diverse range of healthcare diagnostics. The convenience and efficiency of our platform make it particularly advantageous for multi-specialty hospitals and smart diagnostic centers, where streamlined processes and quick, accurate results are crucial. By combining accessibility with affordability, we aim to break down barriers to healthcare and redefine the standards for comprehensive disease detection.Our project not only revolutionizes disease detection but also transforms the accessibility and cost-effectiveness of medical care. By enabling at-home testing and delivering accurate results in a short period, we facilitate a paradigm shift in healthcare delivery. This model is especially beneficial for multi-specialty hospitals and smart diagnostic centers, where our platform's ability to detect a spectrum of diseases efficiently contributes to a more streamlined and effective diagnostic process. As we prioritize user-friendly accessibility and affordability, our project holds the potential to significantly impact healthcare delivery, making quality diagnostics more readily available to a broader demographic.

In the future, as the availability of data continues to increase, there's immense potential to enhance the capabilities of our medical diagnosis application. By incorporating additional disease detections, particularly those detectable through X-ray scans or simple numerical inputs, we can broaden the scope of the application to offer a more comprehensive diagnostic platform. Moreover, extending the functionality to provide tailored advice on precautions and self-care measures for individuals who test positive can significantly improve user engagement and contribute to better health outcomes. Additionally, implementing features to track detection records over time can offer valuable insights for both users and healthcare providers, facilitating proactive management of health conditions. These proposed improvements and modifications promise to make the application more accurate, precise, and user-centric, catering to the evolving needs of the healthcare landscape.

REFERENCES:

1W. Taylor et al., “A review on the state of the art in non contact sensingfor COVID-19,” Sensors, vol. 20, 2020, Art. no. 5665.

2M. Teymouri et al., “Recent advances and challenges of RT-PCR tests for the diagnosis of COVID-19,” Pathol.- Res. Pract., vol. 221, 2021,Art. no. 153443.

3 S. Kadry et al., “Development of a machine-learning system toclassify lung CT scan images into normal/COVID-19 class,” 2020, arXiv:2004.13122.

4L. R. Baltazar et al., “Artificial intelligence on COVID-19 pneumonia detection using chest xray images,” PLoS One, vol. 16, no. 10, 2021,Art. no. e0257884.

5M. Berrimi, S. Hamdi, R. Y. Cherif, A. Moussaoui, M. Oussalah, and M. Chabane, “COVID- 19 detection from xray and CT scans using trans- fer learning,” in Proc. Int. Conf. Women Data Sci. Taif Univ., 2021,pp. 1–6.

6M. L. Holshue et al., “First case of 2019 novel coronavirus in the UnitedStates,” New England J. Med., vol. 382, pp. 929–936, 2020.

7L. Orioli, M. P. Hermans, J.-P. Thissen, D. Maiter, B. Vandeleene, andJ.-C. Yombi, “COVID- 19 in diabetic patients: Related risks and specifics of management,” Annales d’Endocrinologie, vol. 81, no. 2, pp. 101–109,2020.

8P. K. Sethy and S. K. Behera, “Detection of coronavirus disease(COVID-19) based on deep features,” Preprints, vol. 2020030300, 2020,Art. no. 2020

9 S. Basu, S. Mitra, and N. Saha, “Deep learning for screening COVID-19using chest X-ray images,” in Proc. IEEE Symp. Ser. Comput. Intell., 2020,pp. 2521–2527.

10Y. Feng, H. S. Teh, and Y. Cai, “Deep learning for chest radiology: Areview,” Curr. Radiol. Rep., vol. 7, no. 8, pp. 1–9, 2019.