CHRONIC KIDNEY DISEASE IDENTIFICATION USING RANDOM FOREST AND XGBOOST

A PROJECT REPORT

Submitted by

SAYAK DAS [RA2011026010101]

ROOPAL SOOD [RA2011026010103]

Under the Guidance of

Dr. M.S. ABIRAMI
(Associate Professor, Department of Computational Intelligence)

in partial fulfillment of the requirementsfor the degree of

BACHELOR OF TECHNOLOGY
in
COMPUTER SCIENCE AND ENGINEERING
with specialization in Artificial Intelligence and Machine Learning

DEPARTMENT OF COMPUTATIONAL INTELLIGENCE
COLLEGE OF ENGINEERING AND TECHNOLOGY
SRM INSTITUTE OF SCIENCE ANDTECHNOLOGY
KATTANKULATHUR- 603 203

MAY 2024

Department of Computational Intelligence
SRM Institute of Science & Technology
Own Work Declaration Form

Degree/ Course	: B.Tech in Computer Science and Engineering
Student Name	: Sayak Das, Roopal Sood
Registration Number	: RA2011026010101, RA2011026010103
Title of Work	: Chronic Kidney Disease Identification using Random Forest and XGBoost

We hereby certify that this assessment compiles with the University's Rules and Regulations relating to Academic misconduct and plagiarism, as listed in the University Website, Regulations, and the Education Committee guidelines.

We confirm that all the work contained in this assessment is our own except where indicated, and that we have met the following conditions:

- Clearly referenced / listed all sources as appropriate
- Referenced and put in inverted commas all quoted text (from books, web, etc)
- Given the sources of all pictures, data etc. that are not my own
- Not made any use of the report(s) or essay(s) of any other student(s) either past or present
- Acknowledged in appropriate places any help that I have received from others (e.g.fellow students, technicians, statisticians, external sources)
- Compiled with any other plagiarism criteria specified in the Course handbook / University website

We understand that any false claim for this work will be penalized in accordance with theUniversity policies and regulations.

DECLARATION:
We are aware of and understand the University's policy on Academic misconduct and plagiarism and we certify that this assessment is our own work, except where indicated by referring, and that we have followed the good academic practices noted above. RA2011026010101 RA2011026010103
If you are working in a group, please write your registration numbers and sign with the date forevery student in your group.

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR – 603 203

BONAFIDE CERTIFICATE

Certified that 18CSP109L project report titled "**Chronic Kidney Disease Identification using Random Forest and XGBoost**" is the bonafide work of "**SAYAK DAS [RA2011026010101], ROOPAL SOOD [RA2011026010103]**" who carried out the project work under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE	**SIGNATURE**
DR. M.S. ABIRAMI	**Prof. DR. R. ANNIE UTHRA**
SUPERVISOR	**HEAD OF THE DEPARTMENT**
ASSOCIATE PROFESSOR	
DEPARTMENT OF COMPUTATIONAL INTELLIGENCE	DEPARTMENT OF COMPUTATIONAL INTELLIGENCE

Examiner I　　　　Examiner II

ACKNOWLEDGEMENT

We express our humble gratitude to **Dr. C. Muthamizhchelvan**, Vice-Chancellor, SRM Institute of Science and Technology, for the facilities extended for the project work and his continued support. We extend our sincere thanks to Dean-CET, SRM Institute of Science and Technology, **Dr. T.V.Gopal**, for his valuable support.

We wish to thank **Dr. Revathi Venkataraman**, Professor & Chairperson, School of Computing. SRM Institute of Science and Technology, for her support throughout the project work. We are incredibly grateful to our Head of the Department, **Dr. R. Annie Uthra,** Professor, Department of Computational Intelligence, SRM Institute of Science and Technology, for her suggestions and encouragement at all stages of the project work. We are also very thankful to our Project coordinator and Academic Advisor, **Dr. S. Karthick**, Associate Professor, Department of Computational Intelligence for his continuous support throughout our journey.

We want to convey our thanks to our Panel Head and guide, **Dr. M. S. Abirami**, Associate Professor, Department of Computational Intelligence, SRM Institute of Science and Technology, for her input during the project reviews and support. We register our immeasurable thanks to our Faculty Advisor, **Dr. K. Suresh**, Assistant Professor, Department of Computational Intelligence, SRM Institute of Science and Technology, for leading and helping us to complete our course.

We sincerely thank the Computational Intelligence staff and students, SRM Institute of Science and Technology, for their help during our project. Finally, we would like to thank parents, family members, and friends for their unconditional love, constant support, and encouragement.

Sayak Das [Reg No: RA2011026010101]

Roopal Sood [Reg No:RA2011026010103]

ABSTRACT

Globally, chronic kidney disease (CKD) impacts millions of individuals and is a major public health concern, with early identification being crucial for effective management and treatment. In this study, we propose a machine learning-based approach utilizing Random Forest and XGBoost algorithms for the accurate identification of CKD. Clinical and demographic data from both healthy people and patients with CKD make up the dataset used in this investigation. Included are details like blood pressure, serum creatinine levels, age, gender, and other pertinent medical characteristics. To guarantee data quality and consistency, preprocessing methods such as feature scaling, normalisation, and missing value imputation are used. We utilize Random Forest and XGBoost, both renowned ensemble learning algorithms, for the classification task due to their robustness, scalability, and adeptness in handling complex datasets. To enhance model performance, hyperparameter tuning techniques like grid search and cross-validation are implemented. The models' efficacy is assessed using diverse metrics such as accuracy, precision, recall, F1-score, and the area under the receiver operating characteristic curve (AUC-ROC). We conduct a comparative analysis to gauge the effectiveness of Random Forest and XGBoost in identifying CKD. Furthermore, we perform feature importance analysis to pinpoint the most influential predictors contributing to the classification task. The findings demonstrate that both Random Forest and XGBoost models exhibit high accuracy in detecting CKD patients, with XGBoost showing slightly better performance across certain evaluation metrics. Through feature importance analysis, critical clinical indicators essential for distinguishing between CKD and non-CKD cases are identified, offering valuable insights for healthcare professionals. In summary, our machine learning approach utilizing Random Forest and XGBoost algorithms yields promising outcomes in accurately identifying CKD. This research contributes to the expanding realm of utilizing machine learning techniques to enhance healthcare outcomes, especially in the early diagnosis and management of chronic illnesses like CKD. Further validation and integration of these models into clinical practice hold the potential to enhance the efficiency of CKD diagnosis and treatment strategies.

TABLE OF CONTENTS

APPENDIX

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

ANN	Artificial Neural Network
HTML	Hyper Text Markup Language
CSS	Cascading Style Sheet
KNN	K Nearest Neighbour
CKD	Chronic Kidney Disease
IDE	Integrated Development Environment
AHDCNN	Adaptive Hybridized Deep Convolutional Neural Network
XGBoost	eXtreme Gradient Boosting

CHAPTER 1

INTRODUCTION

1.1 Overview

Chronic kidney disease (CKD) is a widely prevalent and critical medical condition distinguished by the progressive deterioration of renal function. Its global impact extends to millions of individuals and is accompanied by substantial healthcare expenditures, morbidity, and mortality. Intervention and early detection are critical for preventing complications and delaying the progression of CKD.

Clinical indicators such blood creatinine levels, Estimated Glomerular filtration rate (EGFR), and proteinuria are frequently used in traditional CKD diagnosis techniques. Even while these markers are useful, it's possible that they don't fully convey the intricacy of CKD progression and the underlying risk factors. Additionally, the subjective and time-consuming manual interpretation of these indicators may result in diagnostic mistakes.

Over the past few years, machine learning methods have emerged as promising tools for refining disease diagnosis and risk assessment, with notable applications in nephrology. Ensemble learning algorithms, such as Random Forest and XGBoost, have showcased their prowess in managing intricate healthcare datasets and extracting valuable insights. This study endeavors to investigate the potential of Random Forest and XGBoost algorithms in identifying CKD using an extensive array of clinical and demographic features. By harnessing the capabilities of these advanced machine learning techniques, our objective is to augment the precision and effectiveness of CKD diagnosis, thereby advancing patient outcomes and healthcare provision.

Progressive renal dysfunction is a defining characteristic of chronic kidney disease (CKD). Cardiovascular illness, diabetes, hypertension, and obesity are common risk factors for chronic kidney disease (CKD). Patients may suffer from symptoms like weariness, electrolyte imbalances, and fluid retention as their kidney function deteriorates. If chronic kidney disease (CKD) is not treated, it can result in problems such renal failure, cardiovascular disease, and early mortality.

In order to limit the progression of the disease and avoid complications, early detection and therapy of CKD are essential. However, because there are no particular symptoms and kidney function is determined by laboratory testing, diagnosing chronic kidney disease (CKD) in its beginning stages can be difficult.

Healthcare professionals can initiate prompt therapies to preserve kidney function and reduce the progression of the disease when chronic kidney disease (CKD) is identified early. This covers comorbid conditions like diabetes and hypertension monitoring, medication management, and lifestyle changes. Furthermore, prompt referral to nephrology specialists for advanced care and renal replacement therapy, including dialysis or kidney transplantation, is made possible by early identification. Even though early detection is crucial, chronic kidney disease (CKD) is frequently misdiagnosed until later on, when symptoms worsen and complications develop. This emphasizes the requirement for accurate and effective CKD screening and diagnosis techniques.

Machine learning techniques have demonstrated significant potential across diverse healthcare realms, encompassing disease forecasting, risk assessment, and optimizing treatment strategies. These methods excel in scrutinizing extensive clinical datasets, unveiling intricate patterns and correlations that may elude human observation. Through the utilization of ML, researchers can construct predictive models for disease detection, prognosis determination, and gauging treatment efficacy, ultimately enhancing patient care and outcomes.

Ensemble learning algorithms, exemplified by Random Forest and XGBoost, stand out as particularly suitable for healthcare contexts. Their adeptness lies in their capacity to manage heterogeneous data, address missing values, and alleviate overfitting concerns. By amalgamating multiple base learners, these algorithms generate predictions that are both robust and precise, thereby facilitating the development of reliable models for healthcare applications. Preprocessing steps, such as addressing missing data, scaling features, and normalizing the dataset, are implemented to ensure data consistency and quality. Techniques like grid search and cross-validation are utilized for hyperparameter tuning, aiming to optimize the models' performance. Various metrics, including accuracy, precision, recall, F1-score, and AUC-ROC, are employed to evaluate the models' performance. Comparative analysis is conducted to assess the efficacy of Random Forest and XGBoost in identifying CKD. Additionally, feature importance analysis is performed to identify the key predictors influencing the classification task.

In summary, this project aims to develop accurate and dependable CKD identification models using Random Forest and XGBoost algorithms. Leveraging machine learning techniques, the goal is to enhance the efficiency of CKD diagnosis and contribute to improved patient outcomes in managing this chronic condition.

1.2 Motivation

The motivation of our major project, “Chronic Kidney Disease Identification Using Random Forest and XGBoost,” is to develop accurate and reliable models for the early detection and diagnosis of chronic kidney disease (CKD). CKD is a prevalent and serious health condition that affects millions of people worldwide, leading to significant morbidity, mortality, and economic burden. For timely intervention and management to impede disease progression and enhance patient outcomes, early detection of CKD is critical. However, diagnosing CKD in its early stages presents challenges due to its asymptomatic nature and the complexity of clinical manifestations.

In this project, we aim to leverage machine learning techniques, specifically Random Forest and XGBoost algorithms, to address these challenges and enhance CKD diagnosis. Random Forest and XGBoost are ensemble learning methods known for their robustness, scalability, and ability to handle complex datasets effectively. By harnessing these algorithms, we seek to develop predictive models capable of accurately identifying CKD patients based on clinical and demographic data.

A comprehensive assessment of the performance of the developed models will be conducted, encompassing a range of metrics such as accuracy, precision, recall, F1-score, and the area under the receiver operating characteristic curve (AUC-ROC). A comparative analysis will be undertaken to evaluate the efficacy of Random Forest and XGBoost in the identification of chronic kidney disease (CKD), thereby offering valuable insights into the merits and drawbacks of each methodology. Additionally, feature importance analysis will be performed to identify the most significant predictors contributing to the classification task. This analysis will help uncover key clinical indicators that play a crucial role in distinguishing between CKD and non-CKD cases, providing valuable insights for medical practitioners and researchers.

Overall, the purpose of this project is to contribute to the advancement of CKD diagnosis by developing accurate and reliable machine learning models using Random Forest and XGBoost algorithms. By improving the efficiency of CKD identification, we aim to facilitate early intervention and management, ultimately leading to better patient outcomes and healthcare resource utilization.

1.3 Problem Statement

The problem statement behind the development of the Chronic Kidney Disease (CKD) identification project using Random Forest and XGBoost algorithms is to address the pressing need for early and accurate detection of kidney cancer subtypes. CKD is a globally prevalent health concern with increasing incidence and high healthcare costs. Early diagnosis and classification of CKD are crucial for patient survival and effective management. However, current research faces the challenge of developing automated tools capable of precisely identifying CKD subtypes. This project aims to leverage machine learning algorithms, specifically Random Forest and XGBoost, to enhance the efficiency and effectiveness of CKD detection at an early stage. By utilizing these advanced algorithms, the goal is to develop a classification system that not only demonstrates high accuracy but also offers cost-effective solutions, thus optimizing resources in healthcare settings and improving patient outcomes.

1.4 Objectives

The objectives of our research work “Chronic Kidney Disease Identification Using Random Forest and XGBoost” are as follows

Exploring and implementing advanced machine learning models to enhance the accuracy and reliability of CKD predictions, ensuring robust performance in diverse healthcare scenarios. Developing data preprocessing environment to process features dimensions and converting categorical data into numerical form improving the efficiency and accuracy of the CKD prediction system.

Implementing multiple machine learning models that not only identifies chronic kidney disease (CKD) but also provides insights also considering individual health data, lifestyle factors, and genetic predispositions.

Improving the efficiency of CKD diagnosis by leveraging machine learning techniques to enable early detection of the disease, thus facilitating timely intervention and management. Conducting comparative analysis to assess the effectiveness of Random Forest and XGBoost algorithms in identifying CKD, providing insights into the strengths and weaknesses of every approach.

By fulfilling these objectives, the project endeavors to provide a significant contribution to the healthcare domain, by applying machine learning methodologies, the timely detection and treatment of chronic renal disease can be significantly enhanced.

1.5 Scope and Applications

The scope and applications of the Chronic Kidney Disease (CKD) identification project using Random Forest and XGBoost algorithms are extensive and impactful in several domains

The research work aims to develop a robust and accurate CKD identification system that can detect the disease at an early stage. This early detection can lead to timely intervention and management, thereby improving patient outcomes and quality of life. By utilizing machine learning algorithms such as Random Forest and XGBoost, the project seeks to accurately classify different subtypes of CKD. This classification can provide valuable insights into disease progression and facilitate personalized treatment strategies for patients.

By reducing the number of superfluous tests and procedures and expediting the diagnostic process, the CKD identification system can optimize healthcare resources. Enhanced resource allocation and cost reductions may result from this efficacy in healthcare contexts.

The research work can contribute to ongoing research efforts and clinical trials focused on CKD by providing a reliable tool for patient stratification and outcome prediction. This can accelerate the development of new therapies and interventions for CKD management.

The CKD identification system can be integrated into telemedicine platforms and remote monitoring devices, allowing for remote diagnosis and monitoring of CKD patients. This enables access to healthcare services in remote areas and enhances patient convenience and accessibility.

The project has implications for public health interventions aimed at reducing the burden of CKD at a population level. By identifying high-risk individuals and implementing targeted prevention strategies, the project can contribute to the prevention and control of CKD on a larger scale.

Overall, the scope and applications of the CKD identification project using Random Forest and XGBoost algorithms extend across clinical practice, research, public health, and healthcare delivery, with the potential to make a significant impact on CKD management and patient care.

1.6 Software Requirements Specification

The software requirements for our research work, “Chronic Kidney Disease Identification Using Random Forest and XGBoost,” encompass several components to facilitate seamless development and integration. We utilize Visual Studio Code (VSCode) as our primary integrated development environment (IDE) for code development, providing a user-friendly and efficient platform for writing, testing, and debugging code. For prediction tasks, various machine learning models will be employed, including Random Forest and XGBoost, implemented using Python’s scikit-learn library. Integration of these models into a web application is facilitated through Python Flask, allowing for seamless communication between the machine learning backend and the front-end interface. The web application’s design and layout are created using HTML and CSS, ensuring a visually appealing and intuitive user experience. By leveraging these technologies and tools, we aim to develop a comprehensive solution for CKD identification that is both robust and user-friendly.

CHAPTER 2
LITERATURE SURVEY

Research work talks about various papers and articles that were published to make improvements regarding this field. Related software talks about the existing application that were built in relation to Chronic Kidney Disease detection.

Guozhen chen et al. [1] addressed the critical need for early detection and classification of kidney cancer, which is crucial for patient survival and management of chronic kidney disease (CKD). It introduces an Adaptive Hybridized Deep Convolutional Neural Network (AHDCNN) aimed at efficiently and effectively identifying kidney disease subtypes. The paper emphasizes the importance of accurate classification methods and the role of data sets in enhancing classification system accuracy. The proposed AHDCNN model utilizes deep learning techniques to extract features from CT images for renal cancer detection. It integrates CNN features with a support vector machine and employs the utilization of fully convolutional networks and conditional random fields (CRFs) in kidney cancer segmentation shows promising outcomes in the early detection and diagnosis of Chronic Kidney Disease (CKD), as demonstrated through experimental procedures conducted on the Internet of Medical Things platform. Additionally, the paper reviews related works in the field of kidney cancer prediction, highlighting various machine learning approaches such as Neighborhood Component Analysis (NCA), Deep Neural Network (DNN), Hybrid Neural Network (HNN), and Recurrent Neural Network (RNN).In summary, the research proposes an innovative approach using deep learning techniques for the prediction and diagnosis of chronic kidney disease. It demonstrates the potential of machine learning algorithms in improving early detection and management of kidney cancer, leveraging advancements in medical technology and data analytics.

Maithili Desai et al. [2] presented The research paper explores the prevalence and detection of Chronic Kidney Disease (CKD) using data mining techniques, specifically focusing on the Boruta algorithm.The Boruta algorithm, applied in this study, aids in identifying significant factors associated with CKD prediction. It works by creating shadow attributes and training a random forest classifier to assess attribute importance. Out of 24 attributes, only 7 were confirmed to be important in predicting CKD. The analysis shows that reducing the number of features can lead to a slightly lower accuracy of 99.19%, compared to 100% accuracy with all features, but significantly reduces processing time and memory load. The research discusses the importance of factors such as hypertension, blood pressure, and specific tests like urine albumin and serum creatinine in detecting CKD. It emphasizes that a combination of tests is necessary for accurate diagnosis, especially in senior patients who are at higher risk. Furthermore, it suggests potential correlations between age and CKD-related factors.Numerical readings include statistics on attribute importance, with variables like sodium, age, packed cell volume, and hemoglobin deemed significant. Sensitivity and specificity of the model are reported as 1 and 0.98 respectively, with an out-of-bag estimate error rate of 1.08%. Confusion matrices illustrate the model's performance, with 47 correctly classified as "notCKD" and 75 as "CKD" in the reduced feature model. In conclusion, the study underscores the economic burden of CKD and the importance of affordable diagnostic methods. Boruta Analysis emerges as a valuable tool for medical diagnosis, offering cost-effective and faster solutions. The paper suggests avenues for future research, including the application of data mining algorithms in other chronic diseases for early detection and improved patient outcomes.

Pedro A. Moreno-Sánchez et al. [3] highlighted the significance of leveraging advanced technologies like deep learning and AI models to enhance diagnostic accuracy. These studies introduce innovative approaches such as adaptive hybridized deep convolutional neural networks (AHDCNN), explainable AI models, and multi-class kidney abnormalities detection systems, aimed at efficiently identifying CKD subtypes and enabling timely interventions. They underscore the importance of transparent, interpretable AI models in facilitating informed clinical decision-making and improving patient management strategies.
Moreover, beyond early detection, prognostic assessments hold paramount importance in guiding treatment decisions and optimizing patient outcomes.

Ping Liang et al. [4] conducted studies on the identification of intelligible predictors of poor prognosis in CKD. These investigations underscore the critical need for developing clinically applicable machine learning approaches that not only enhance prognostic accuracy but also cater to low-cost diagnostic screening, thereby improving accessibility to essential healthcare services. By harnessing deep learning models and leveraging comprehensive clinical datasets, these efforts aim to improve risk stratification and enable targeted interventions tailored to individual patient needs. This holistic approach holds immense promise in optimizing resource allocation and enhancing the overall quality of CKD management, ultimately benefiting patient outcomes on a global scale.

Asif Salekin et al. [5] researched on the efficacy of various machine learning algorithms for CKD detection. These studies emphasize the importance of selecting robust algorithms capable of reliably identifying CKD patterns and differentiating between disease subtypes. Through rigorous evaluation of different models, researchers gain valuable insights into the strengths and limitations of each approach, paving the way for the development of more accurate and efficient diagnostic tools. This iterative process of refinement is essential for advancing the field of CKD detection and ensuring the delivery of high-quality care to patients worldwide.

Dina Saif et al. [6] were focused on enhancing the accessibility of CKD screening by developing deep learning frameworks and hybrid models. These studies aim to address disparities in healthcare access by providing cost-effective and scalable diagnostic solutions. Leveraging machine learning techniques, researchers strive to democratize access to CKD screening, empowering healthcare providers with actionable insights for early detection and prevention. This approach is particularly crucial in underserved communities where access to specialized healthcare services may be limited, underscoring the transformative potential of machine learning in improving health outcomes and reducing health disparities on a global scale.

Sagar Dhanraj et al. [7] addressed the urgent need for efficient artificial intelligence (AI) technology-based kidney anomaly diagnosis in light of the worldwide shortage of nephrologists and the rising prevalence of renal illnesses. A significant percentage of people globally suffer from chronic kidney disease (CKD), a serious health issue. Renal failure can be avoided by detecting kidney diseases, such as cysts, tumors, and stones, early on. Treatment is sometimes postponed since these issues do not always show symptoms straight away. To tackle this problem, the paper presents YOLOv8, a deep learning model trained on a large dataset of 12,446 annotated CT whole abdomen and urogram images, which accentuates kidney abnormalities. The dataset was divided into four categories: cyst, tumor, stone, and normal. The data was supplied by Bangladeshi hospitals in Dhaka, and it was meticulously annotated for model training. The YOLOv8 model demonstrated better accuracy in detecting renal diseases, as demonstrated by its accuracy rate of 82.52%, precision of 85.76%, recall of 75.28%, F1 score of 75.72%, and specificity of 93.12%. Evaluation parameters such as accuracy, precision, recall, F1 score, and specificity showed how well the model identified kidney problems. The study presents a revolutionary platform for the clinical identification of renal illnesses, providing medical professionals with a fast and accurate way to diagnose kidney stones, tumors, and cysts through a customizable solution. By using deep learning techniques to improve and automate the detection of kidney problems, this work contributes to the fast growing field of AI-driven healthcare. Using large-scale medical imaging datasets and state-of-the-art AI models like YOLOv8, the research demonstrates how AI may revolutionize kidney disease detection, improving patient outcomes and reducing workload for medical practitioners.

MD. Rashed Al Mahfuz et al. [8] focused on developing machine learning models for early identification of Chronic Kidney Disease (CKD) in order to address the challenges associated with late-stage diagnosis and limited testing facilities, particularly in less developed places. Early CKD detection can significantly improve patient outcomes and care while saving money on diagnosis. The study employed certain pathological categories to identify important clinical test criteria for a precise diagnosis of chronic kidney disease (CKD), with the aim of optimizing diagnostic screening processes. Numerous machine learning classifiers were evaluated by utilizing optimum datasets produced by k-fold cross-validation from these selected attributes. The results demonstrated how well these improved datasets performed in the diagnosis of chronic kidney disease, including clinical variables and major clinical test characteristics from blood and urine tests. It was demonstrated that the Random Forest (RF) classifier was the most effective model due to its high accuracy in diagnosing CKD. The study demonstrates the potential of machine learning-based techniques for precise and reasonably priced CKD screening, which would allow for early intervention and treatment planning. Through the identification and utilization of critical diagnostic characteristics, this research contributes to the development of an innovative tool for improved CKD screening, particularly valuable in healthcare settings with limited resources. The research highlights the necessity of machine learning models that can be interpreted in order to diagnose chronic kidney disease (CKD). By evaluating model results and understanding the importance of specific traits in decision-making processes, these models help doctors improve patient care and diagnostic precision.

Rahul Sawhney et al. [9] addressed the use of a deep neural network-based Multi-Layer Perceptron (MLP) Classifier for kidney failure diagnosis—a major side effect of chronic kidney disease (CKD) that impacts a significant portion of the global population—is examined. It is difficult to detect chronic kidney disease (CKD) early since there are usually no symptoms until significant kidney damage has developed. This emphasizes the critical need for a correct diagnosis as soon as possible. The proposed MLP classifier was trained utilizing a range of symptoms and indications, such as age, blood sugar levels, and red blood cell count, using a dataset of 400 individuals. The experimental results demonstrated that the MLP classifier achieved perfect testing accuracy in recognizing CKD, outperforming more well-known machine learning models like support vector machines and naive Bayes classifiers.

The study emphasizes the promise of deep learning techniques, such as neural networks, in healthcare applications, such as the detection of chronic kidney disease. Neural networks are particularly good at handling complex and nonlinear data patterns, which is useful for properly categorizing illnesses. Using the PyTorch library, the article demonstrates the design and training process of the MLP classifier, highlighting the classifier's ability to learn from datasets and change on its own. This adaptability is crucial since each person's CKD will manifest differently and progress at a different rate. The necessity of early CKD detection is underscored by the disease's substantial global economic and health consequences. Prompt care is essential to avoid end-stage renal disease (ESRD), which requires costly treatments such as dialysis or kidney transplantation since chronic kidney disease (CKD) progresses slowly and presents with mixed symptoms. The paper also discusses the limitations of the diagnostic methods now in use, highlighting the potential for artificial intelligence (AI) and neural networks to improve accuracy and efficacy. A number of studies examining the application of neural networks and machine learning for the detection and treatment of chronic renal disease are also highlighted in the paper's literature analysis. These include developing AI-powered tools for patient monitoring and treatment planning, assessing cardiovascular risk using decision-based neural networks, and estimating CKD stage by image segmentation.

Dr. Razib Hayat Khan et al. [10] discussed on the aims of appropriately forecast chronic kidney failure (CKF) by the application of machine learning (ML) techniques, therefore addressing the rising occurrence of CKF worldwide. To maximize accuracy and address issues like overfitting, the study employs a range of machine learning techniques, such as XGBoost, Random Forest, AdaBoost, Logistic Regression, and a special Hybrid Model. The first component of the paper discusses the value of accurate prediction models in healthcare, particularly in the identification of chronic disorders like CKF. Chronic kidney failure has a significant health cost, which highlights the need for effective early detection and treatment programs, especially in low- and middle-income countries. Some of the important methodologies addressed are preprocessing the dataset, selecting features using Pearson correlation, and building the hybrid model using ensemble techniques like stacking. Each machine learning technique is covered in detail, with an emphasis on its benefits and its applications in the field of medical diagnostics. The Hybrid Model achieves 94% accuracy, 96.4% precision, and 96.8% recall, outperforming standalone algorithms like Random Forest and XGBoost. This outstanding achievement is highlighted by the outcomes and justifications. The study demonstrates the Hybrid Model's ability to effectively balance accuracy and memory, which is crucial for accurate sickness prediction.

The study clarifies the application of ML in healthcare, namely in the prediction of long-term conditions like congestive heart failure. By combining several machine learning algorithms into a hybrid model, the research demonstrates increased accuracy and robustness, which is important for precise sickness prognosis and timely medical interventions.

Md. Ariful Islam et al. [11] researched to develop an algorithm for identifying Chronic Kidney Disease (CKD) through the use of machine learning. The dataset employed comprises 400 instances, of which 14 are nominal and 11 are numerical features totaling 24. Encoding categorical variables and managing absent values are components of data preprocessing. Machine learning models such as Gradient Boosting, Decision Tree, XGBoost, CatBoost, KNN, and Random Forest are trained and assessed utilizing performance metrics including F1-score, accuracy, precision, recall, and recall. Preprocessing data, identifying pertinent features, and training models with various algorithms are critical stages. A 10-fold cross-validation technique is implemented in order to verify the models and mitigate the risk of overfitting. Evaluation of performance reveals that after implementing PCA, XGBoost achieved the highest accuracy of 99.2%, followed by AdaBoost, Random Forest, and Gradient Boosting. The research highlights the significance of feature selection and model optimization in enhancing the precision of predictions. The outcomes exhibit encouraging performance in the detection of chronic kidney disease (CKD), as models proficiently employ characteristics such as serum creatinine, specific gravity, hemoglobin, and others. The limited size of the dataset and the requirement for more representative data for model training and generalization are both limitations. Although the proposed model has these constraints, it has the potential to aid medical professionals in the early detection and risk assessment of chronic kidney disease. The research concludes with a comprehensive machine learning approach to CKD identification, emphasizing the importance of feature selection, data preprocessing, and model evaluation. It is recommended that additional research be conducted in order to improve the performance of models using datasets that are more extensive and varied. This would ultimately enhance the precision and dependability of CKD diagnosis in clinical environments.

Dibaba Adeba Debal et al. [12] conducted their research at St. Paulo's Hospital in Ethiopia was to apply machine learning methodologies to the prediction of chronic kidney disease (CKD). There were 1718 instances in the dataset, each containing 19 characteristics, such as numeric and nominal values. In order to prepare the dataset and resolve missing values, preprocessing procedures including normalization, handling missing values, and outlier removal were implemented. In order to ascertain pertinent predictive features, feature selection techniques such as Recursive Feature Elimination with Cross-Validation (RFECV) and Univariate Feature Selection (UFS) were applied. Binary and five-class classification assignments were executed utilizing three machine learning algorithms: Random Forest, Support Vector Machine (SVM), and Decision Tree (DT). In addition to tenfold cross-validation, several performance metrics—including accuracy, precision, recall, F1-score, sensitivity, and specificity—were employed to assess the models. Prior to feature selection, Support Vector Machines (SVM) demonstrated the highest accuracy for binary classification at 99.8%, whereas XGBoost attained an accuracy of 82.56% for five-class classification. Following feature selection, XGBoost outperformed SVM and RF with RFECV for the five-class dataset, while SVM and RF with RFECV demonstrated the highest accuracy for the binary class. The results of the study suggest that SVM, RF with RFECV, and XGBoost have the potential to be implemented in clinical settings for the prediction of CKD. Medical professionals may be able to make quicker and more precise diagnoses with the assistance of these models, which may improve patient outcomes. Subsequent avenues of inquiry may involve the implementation of unsupervised or deep learning algorithms, as well as the creation of mobile-based systems that enable continuous monitoring of patients with CKD.

Hira Khalid et al. [13] focused on making an accurate diagnosis of chronic kidney disease (CKD) by utilizing machine learning techniques. A hybrid model is suggested in which feature selection is performed using Pearson correlation. A range of machine learning classifiers are assessed in the study, such as gradient boosting, random forest, Gaussian Naïve Bayes, and decision tree. The random forest meta-classifier is integrated with these classifiers to form the hybrid model, which achieves one hundred percent accuracy on the chronic renal disease dataset from UCI. This research examines established classification methodologies, elucidates obstacles encountered, and juxtaposes accuracy metrics. The hybrid model outperforms the random forest (98%), decision tree (96%), and gradient boosting (91%). Naive Bayes, decision tree, K-nearest neighbor, random forest, support vector machine, LDA, GB, and neural network are the primary algorithms used for CKD prediction.

Imesh Udara et al. [14] conducted studies centers on the application of machine learning techniques for the prediction of chronic kidney disease (CKD). CKD is a substantial worldwide health issue, resulting in millions of fatalities each year. Although conventional diagnostic approaches are accessible, this research utilizes machine learning on account of its capacity to deliver exceptional precision and efficacy. For the prediction of CKD, the research paper proposes a hybrid model that incorporates multiple machine learning techniques. The Pearson correlation method is employed during feature selection in order to ascertain pertinent predictors from the dataset. The selected models comprise gradient boosting, Gaussian Naïve Bayes, random forest, decision tree classifier, and random forest. In the composite model, random forest functions as the meta-classifier. The models are assessed in terms of accuracy with the objective of determining the most efficient classifier for predicting CKD. The research compares and contrasts the accuracy scores of extant machine learning classification techniques in a critical manner. In CKD prediction, Naive Bayes, decision trees, K-nearest neighbors, support vector machines, LDA, GB, and neural networks are notable algorithms. The study constructs a hybrid model to forecast chronic kidney disease (CKD) by individually implementing the top four models from the UCI chronic kidney disease dataset. Accuracy rates of approximately 99% were attained by gradient boosting, 98% by random forest, and 96% by decision tree classifier. It is noteworthy that the hybrid model that has been proposed attains a flawless accuracy rate of one hundred percent when applied to the identical dataset, thus showcasing its exceptional performance. By demonstrating the efficacy of machine learning in CKD prediction and proposing a hybrid model that outperforms individual classifiers, the research makes a contribution to the field.

Deema Mohammed Alsekait et al. [15] introduced an innovative ensemble deep learning (DL) methodology for the detection of chronic kidney disease (CKD), which makes use of feature selection techniques to ascertain pertinent features from the dataset. The dataset, obtained from the machine learning repository at UCI, comprises 24 features that have been classified into 11 numeric and 13 categorical categories, with each category being assigned a class label that denotes positive or negative chronic kidney disease. Encoding categorical features and applying statistical methods to address missing values and outliers are all components of data preprocessing. Filter, wrapper, and embedded feature selection methods, as well as the chi-squared test, recursive feature elimination (RFE), mutual information, and tree-based approaches, are investigated. The model under consideration consists of two tiers: a base learning tier utilizing optimized DL models (RNN, GRU, LSTM); and a metalearner tier incorporating SVM. In order to refine hyperparameters, optimization techniques such as Keras-Tuner and grid search with cross-validation are implemented. Achieving high levels of accuracy, precision, recall, and F1 scores, experimental results illustrate the performance of DL models and the proposed ensemble model when various feature selection methods are implemented. The study further examines the significance of particular features in the prediction of chronic kidney disease (CKD), emphasizing the global and local explainability of the model's decisions. In conclusion, medical insights and comparisons to existing literature provide validation for the efficacy of the proposed method in detecting CKD.

Qiong Bai et al. [16] used machine learning (ML) techniques, the study sought to construct a prediction model for end-stage kidney disease (ESKD) in patients with chronic kidney disease (CKD). As predictors, patient characteristics, medical history, clinical parameters, and blood tests were gathered. The dataset comprised 748 participants who were followed for a mean of 6.3 ± 2.3 years. The majority of patients exhibited CKD stages 2 or 3 at the outset, with 70 patients (9.4%) presenting with ESKD; all of these patients underwent kidney replacement therapy. Utilizing five machine learning algorithms—logistic regression, naïve Bayes, random forest, decision tree, and K-nearest neighbors—grid search was utilized to optimize the hyperparameters of each algorithm. Metrics including accuracy, precision, recall, specificity, F1 score, and area under the curve (AUC) were employed to evaluate the performance of the model. As a benchmark, the Kidney Failure Risk Equation (KFRE) was also compared. Encoding categorical variables with one-hot encoding and managing missing data with five repetitions of multiple imputation constituted the data preprocessing steps. In each subset, stratified cross-validation ensured a balanced distribution of outcome classes. Random forest had the highest area under the curve (AUC) among the ML models, at 0.81, closely followed by logistic regression, naïve Bayes, and KFRE. Contrary to expectations, KFRE exhibited superior accuracy, specificity, and precision in addition to a comparable AUC, despite its simplicity as a three-variable model. At the default threshold, however, its sensitivity was lower than that of some ML models. The insufficient number of urine variables (due to the limited availability of testing) and the requirement for external validation and the incorporation of supplementary clinical characteristics in subsequent investigations were all limitations. Finally, the research showcased the viability of machine learning (ML) in forecasting end-stage renal disease (ESKD) among patients with CKD and emphasized the capacity of ML models to supplement conventional regression techniques in predicting disease risk. Subsequent investigations ought to prioritize the validation and enhancement of models through the utilization of more extensive datasets and a wider range of clinical attributes.

Nikhila et al. [17] employed machine learning (ML) model that could accurately predict chronic kidney disease (CKD) was the objective of this research. The analysis utilized the Indians CKD dataset, which consisted of 400 instances and contained 24 numeric and nominal attributes, obtained from the machine learning repository at UCI. By following a methodical sequence comprising ten stages, the study thoroughly examined diverse facets of data preprocessing, model training, and evaluation. Preliminary stages encompassed the acquisition and examination of datasets, succeeded by the management of absent data and normalization to guarantee consistent attribute scaling. Following this, the dataset was partitioned into separate training and testing sets in order to facilitate the process of model development and validation. On the basis of their efficacy, multiple ML algorithms, including SVM, KNN and AdaBoost, were ranked after being trained on the normalized dataset. A methodology was suggested that employed weighted average ensemble models to aggregate predictions from various base learners: an ensemble approach. The effectiveness of the ultimate model was assessed utilizing metrics such as precision, responsiveness, particularity, and F1 score. The results demonstrated that the ensemble model outperformed individual machine learning algorithms. Considering computational and structural complexity throughout model training and testing, the study concluded that the proposed model provides a dependable and effective solution for rapid and precise CKD screening.

Reshma S et al. [18] researched centers on the application of Support Vector Machine (SVM) for classification and Ant Colony Optimization (ACO) for feature selection in order to forecast Chronic Kidney Disease (CKD). Encoding categorical attributes, managing absent values, and converting data types are all components of data pre-processing. ACO optimizes the process of feature selection by emulating the navigational strategies employed by actual ants when traversing graphs. A supervised learning model, SVM, distinguishes between CKD and non-CKD categories of data. Precision, recall, F1-score, and accuracy are evaluation metrics that illustrate the efficacy of the method. With an approximate 96% accuracy rate, the study demonstrates that ACO and SVM are effective in predicting CKD with a reduced number of attributes and still maintain a high level of accuracy.

Marwa Almasoud et al. [19] dedicated to the preparation of a dataset for the prediction of chronic kidney disease (CKD). This will be achieved by employing machine learning algorithms for feature selection, data preprocessing, and model evaluation. Particularly in medical datasets such as CKD, outliers must be carefully considered. Outliers that are valid and essential for precise diagnosis are preserved, whereas those that indicate errors are regarded as lacking data. Medical datasets often contain missing values, which are remedied by employing multiple imputation methods that rely on logistic and linear regressions. To eliminate redundant attributes, data reduction techniques such as feature associations and correlations are utilized. The elimination of nine redundant features optimizes the dataset for modeling purposes. Further, in order to prepare categorical variables for machine learning algorithms, data transformation encompasses normalization and surrogate encoding. For the detection of CKD, four machine learning algorithms are implemented and assessed: logistic regression, support vector machines (SVM), random forest, and gradient boosting. The algorithm that attains the highest performance is gradient boosting, which obtains 93.1% F1-measure, 97.8% sensitivity, and 96.3% specificity. Notably, the study highlights both the significance of hemoglobin in CKD prognosis and the constraints of the limited dataset size. Subsequent investigations seek to authenticate the results using more extensive datasets and examine predictive frameworks that can identify risk factors for chronic kidney disease (CKD) in individuals who also have diabetes, hypertension, and a familial predisposition to renal failure. Such efforts will ultimately aid in the development of strategies for the early detection and prevention of CKD.

Chamandeep Kaur et al. [20] employed feature selection, the research employed the Ant Colony Optimization (ACO) algorithm to prioritize dataset attributes. Predictions were made using decision trees, random forest, and KNN classifiers, with random forest exhibiting the least feature bias. Although certain characteristics of CKD may be overrepresented, precise prediction may result in false positives. The veracity of missing data replacement through collaborative imputers may be enhanced. Random forest classifiers have the potential to decrease the quantity of necessary medical tests through the process of variable combination and absent value filling. This method emphasizes the significance of domain expertise in CKD detection clinical data interpretation, suggesting that future research should focus on missing value management and the incorporation of additional data sources.

Md. Mehedi Hassan et al. [21] For feature selection, the research employed the Ant Colony Optimization (ACO) algorithm to prioritize dataset attributes. Predictions were made using decision trees, random forest, and KNN classifiers, with random forest exhibiting the least feature bias. Although certain characteristics of CKD may be overrepresented, precise prediction may result in false positives. The veracity of missing data replacement through collaborative imputers may be enhanced. Random forest classifiers have the potential to decrease the quantity of necessary medical tests through the process of variable combination and absent value filling. This method emphasizes the significance of domain expertise in CKD detection clinical data interpretation, suggesting that future research should focus on missing value management and the incorporation of additional data sources. A predictive model is put forth in this study to facilitate the early detection of Chronic Kidney Disease (CKD). To achieve this, machine learning algorithms are utilized, such as Random Tree (RT), Neural Network (NN), Support Vector Machine (SVM), and Bagging Tree Model (BTM). Data preprocessing consists of clustering to identify distinct data groups and the application of Predictive Mean Matching (PMM) to handle absent values. A feature selection algorithm based on XGBoost is employed to extract seven crucial features from the dataset. In addition to the designated features subset, the entire dataset is utilized to train models; the data is then divided into training and testing sets. The findings suggest that NN outperforms SVM on the XGBoost dataset, attaining a remarkable accuracy of 95%. Kappa scores, sensitivity, and specificity differ between algorithms and feature sets. On the complete dataset, NN, RF, and SVM exhibit the highest performance, whereas on the XGBoost subset, SVM surpasses all other models. An assessment in comparison to established systems reveals that competitive accuracy levels are evident. However, there are constraints that arise due to the magnitude of the dataset. Subsequent investigations seek to implement the established models on additional disease datasets while augmenting the sophistication of the system. In its entirety, the research paper introduces a positive strategy for the early detection of chronic kidney disease (CKD), underscoring the criticality of feature selection methods and machine learning algorithms in enhancing diagnostic precision.

All things considered, the study highlights the growing importance of machine learning (ML) in healthcare and provides a comprehensive framework for developing accurate prediction models, which might be helpful for diagnosing illnesses early and providing tailored patient care.

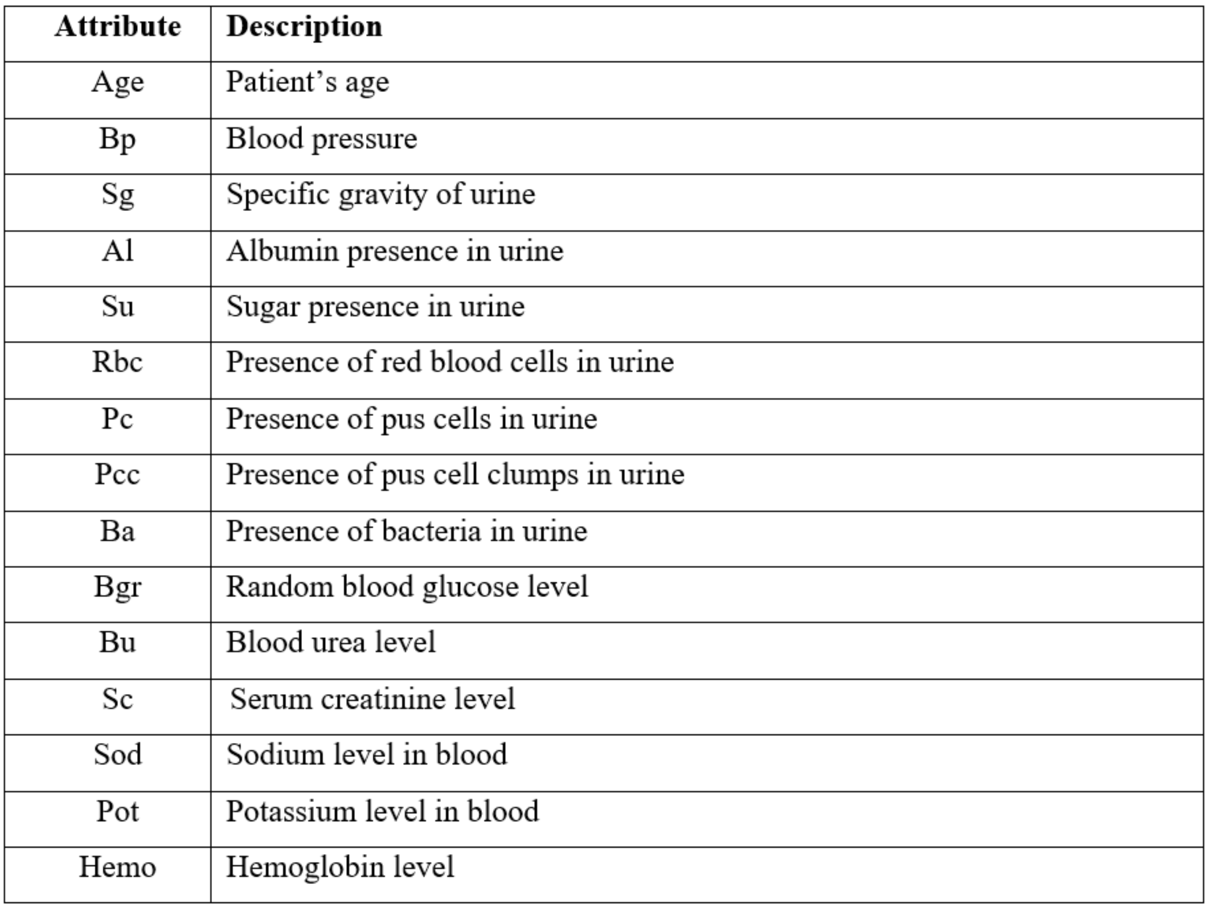
CHAPTER 3
METHODOLOGY OF CHRONIC KIDNEY DISEASE IDENTIFICATION

The methodology for building a machine learning-based system involves several key steps. Firstly, data collection and preprocessing are performed to ensure data quality and consistency. Next, suitable machine learning algorithms are selected and trained using the preprocessed data. Hyperparameter tuning and model evaluation techniques are then employed to optimize performance. Finally, the trained models are integrated into the project's architecture, often utilizing frameworks like Flask for web applications, to deliver a user-friendly and efficient solution.

3.1 Dataset Description

The dataset which is used for this research work consists of 400 rows of data in which every row have 24 features and 1 label . The dataset is collected from Kaggle. Label is categorized into two classes ckd and notCkd. All features are basically human body vital which are directly and indirectly related for the cause of Chronic Kidney Disease. So each data is important for the early prediction.

Table 3.1: Attributes of Kidney Dataset



Attribute	Description
Age	Patient's age
Bp	Blood pressure
Sg	Specific gravity of urine
Al	Albumin presence in urine
Su	Sugar presence in urine
Rbc	Presence of red blood cells in urine
Pc	Presence of pus cells in urine
Pcc	Presence of pus cell clumps in urine
Ba	Presence of bacteria in urine
Bgr	Random blood glucose level
Bu	Blood urea level
Sc	Serum creatinine level
Sod	Sodium level in blood
Pot	Potassium level in blood
Hemo	Hemoglobin level

Pcv	Packed cell volume
Wc	White blood cell count
Rc	Red blood cell count
Htn	Presence of hypertension
Dm	Presence of diabetes mellitus
Cad	Presence of coronary artery disease
Appet	Appetite status
Pe	Presence of pedal edema
Ane	Presence of anemia
classification	CKD class (label)

3.2 Module Description

The research work focuses on identifying chronic kidney disease through an advanced methodology blending machine learning models. The methodology commences with comprehensive data collection, cleaning, and augmentation, followed by the division of the dataset into training, validation, and test sets. Feature extraction is performed using XGBoost, Random Forest and other models tailored for kidney disease identification are developed. The model is trained with advanced machine learning techniques, and advanced machine learning models like XGBoost , Forest tree classifier are explored. Ensemble methods and IoT platform experiments contribute to a holistic evaluation, with metrics such as accuracy, precision, recall, and F1 score. Results are analyzed, guiding fine-tuning and optimization for the development of an efficient and accurate model. Comprehensive documentation and reporting encompassing preprocessing, model architectures, and findings conclude the project. This methodology aims to advance early chronic kidney disease detection, emphasizing the ability of machine learning in healthcare.

3.2.1 Hybrid Machine Learning Model for Identification

The dataset which is used for this research work consists of 24 features and 1 label. All features are basically human body vital which are directly and indirectly related for the cause of Chronic Kidney Disease. Label is categorized into two classes ckd and notCkd. The code begins by importing necessary libraries for data visualization, including pandas, numpy, matplotlib, seaborn, and plotly. It also suppresses warning messages and sets a specific plotting style. Then, the code mounts Google Drive to access data. After loading the dataset into a DataFrame (assumed as `df`), it converts specific columns to numerical type using `pd.to_numeric` with the `errors='coerce'` parameter to handle any conversion errors gracefully.

Categorical and numerical columns are then separated into two lists using list comprehensions. The code iterates over categorical columns to print unique values for each column. This step is useful for understanding the nature and diversity of categorical data in the dataset. Overall, the code sets up the environment, prepares the dataset for analysis, and provides insights into the categorical data's uniqueness, which is crucial for subsequent analysis and visualization tasks.

$$f(x) \approx f(a) + f'(a)(x-a) + \frac{1}{2}f''(a)(x-a)^2 \qquad \text{-------} \quad (1)$$

$$\mathcal{L}^{(t)} \simeq \sum_{i=1}^{n}[l(y_i, \hat{y}^{(t-1)}) + g_i f_t(\mathbf{x}_i) + \frac{1}{2}h_i f_t^2(\mathbf{x}_i)] + \Omega(f_t) \qquad \text{-------} \quad (2)$$

The top equation shows the Taylor Series expansion of the function f(x) around a base point a. This is a way of approximating a function with a polynomial centered around a specific input value. The bottom equation shows part of the XGBoost objective function. In gradient boosting, the goal is to trim down a loss function over a set of training data. The loss function measures how different the predictions of a model are from the actual values. The XGBoost objective function combines a loss function, represented by l(yi,y ^(t−1)) (where i indexes the training sample, yi is the true value for that sample, and y^(t−1) is the prediction from the model at the previous iteration), with a regularization term, represented by Ω(ft). The regularization term penalizes the complexity of the model to prevent overfitting.

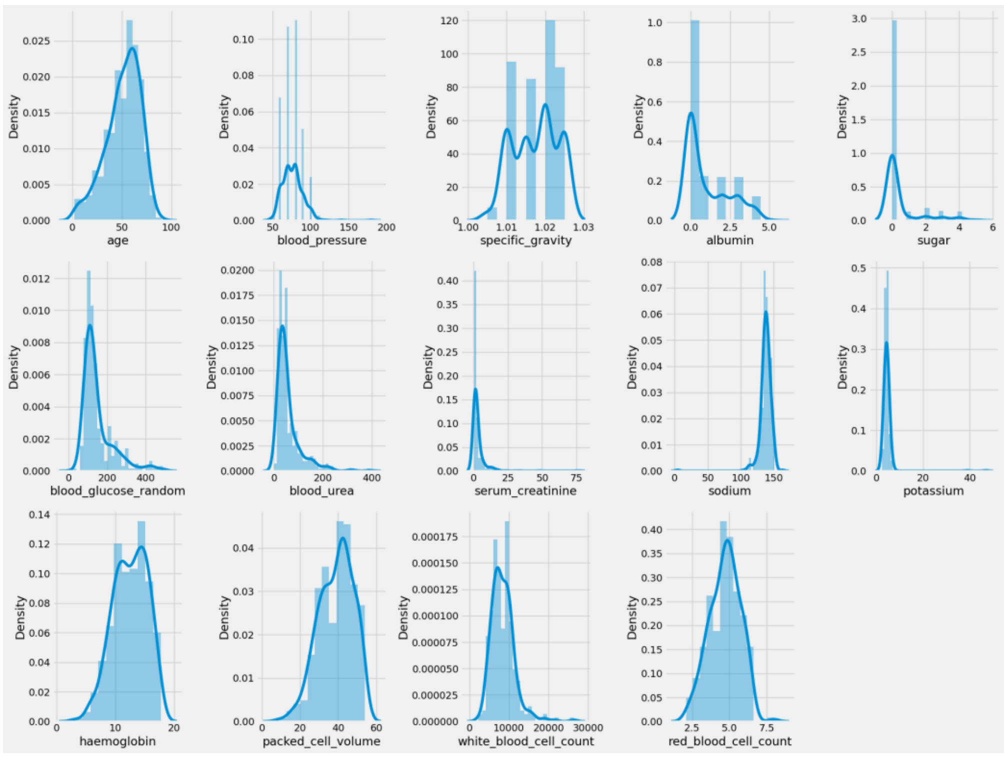


Fig 3.1: Distribution of Numerical Features

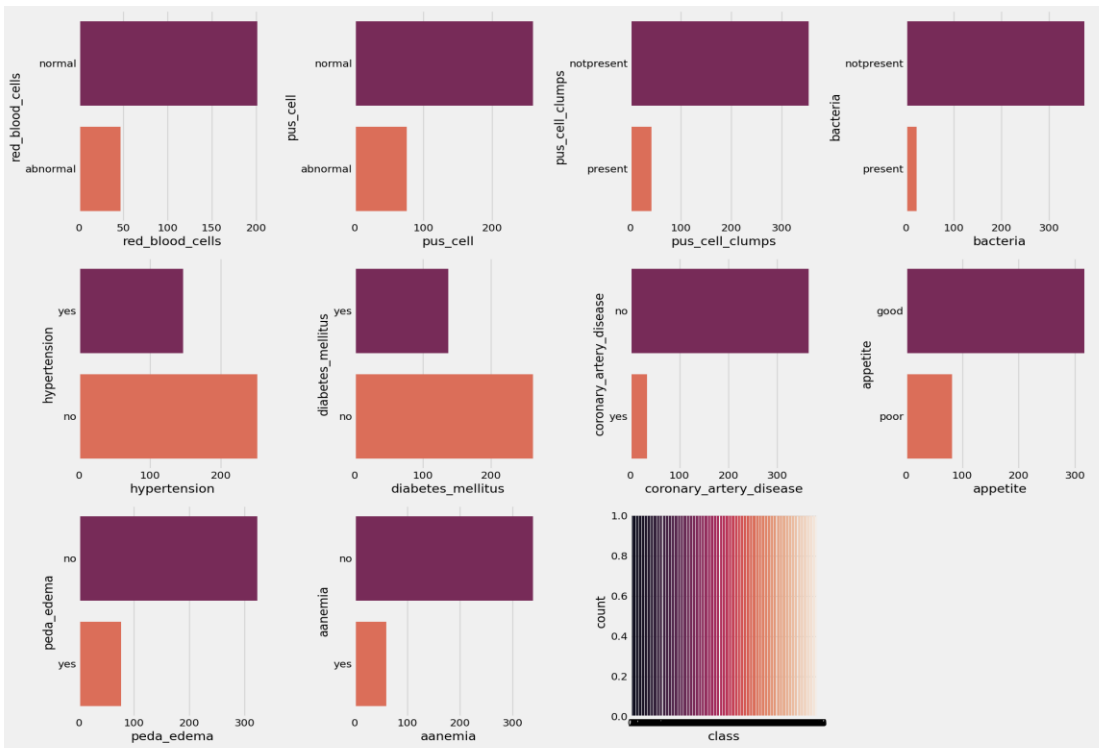


Fig 3.2: Distribution of Categorical Features

The code first creates subplots for visualizing the distribution of numerical and categorical columns separately. For numerical columns, it iterates over each column, plotting a distribution plot using Seaborn's `sns.distplot`. Similarly, for categorical columns, it plots a count plot using `sns.countplot`. The plots are arranged in a grid structure to fit the figure appropriately. Afterward, a heatmap of the correlation matrix is generated using Seaborn's `sns.heatmap`, annotating the correlations and displaying them color-coded. The heatmap is saved as a PNG file named 'corr1.png'. Next, the code checks for missing values in the DataFrame using `df.isna().sum().sort_values(ascending=False)` to identify the columns with null values. It then defines two functions for imputing missing values `random_value_imputation` for numerical columns and 'red_blood_cells' and 'pus_cell' in categorical columns, and `impute_mode` for the rest of the categorical columns. Random sampling is used to fill null values in numerical columns, while mode imputation is used for categorical columns. The missing values are replaced accordingly.

Label encoding is applied to categorical columns using Scikit-learn's `LabelEncoder`, converting categorical values into numerical labels. Finally, correlations with the target variable 'class' are computed and displayed for both numerical and categorical features. This comprehensive approach ensures the dataset is prepared for further analysis and modeling by handling missing values and encoding categorical features appropriately.

The code begins by importing necessary libraries for scaling features, splitting data, and implementing machine learning algorithms. It uses `MinMaxScaler` from Scikit-learn to scale up the features to a specified range. The scaled features are stored in `new_features` after transforming the original feature matrix `X`. The data is then split into training and test sets using `train_test_split` from Scikit-learn, with 75% of the data used for training and 25% for testing, maintaining consistency with a random state of 67 for reproducibility.

Five classification models are trained and evaluated K-Nearest Neighbors (KNN), Random Forest Classifier, Decision Tree Classifier, Gradient Boost Classifier and XGBoost. For each model, the training and test accuracies are printed along with a classification report, which provides metrics like precision, recall, and F1-score for each class. Additionally, confusion matrices are plotted using a custom function `plot_confusion_matrix`. After fitting each model to the training data, the accuracies and classification reports for the test data are printed out. These metrics provide insights into how precise each model performs on unseen data. The final step involves displaying the confusion matrices for each model, allowing for a visual examination of the model's performance in classifying different classes.

A basic machine learning approach called K-Nearest Neighbours (KNN) is applied to both regression and classification problems. Particularly for small to medium-sized datasets, its popularity stems from its simplicity, interpretability, and effectiveness. KNN does not learn a particular model during training; instead, it falls under the category of lazy learning algorithms. Rather, it learns the entire training dataset by heart and predicts new data points by comparing them to the current set. There are multiple essential steps in the KNN operating process. First of all, it needs a training dataset made up of target values for regression tasks or feature vectors with the labels that correspond to them for classification tasks. Furthermore, KNN depends on a parameter called "K," which denotes the quantity of closest neighbours that are taken into account throughout the prediction process. Once initialised, a distance metric—typically the Euclidean distance—is used by KNN to determine the similarity between data points. Depending on the type of data, other distance metrics such the Manhattan distance, Minkowski distance, or cosine similarity may also be used. After the distances are computed, the algorithm determines the 'K' closest neighbours of a given data point in the training set. KNN uses a majority voting process among its 'K' nearest neighbours to determine the class label to be applied to the unseen data point in classification tasks. This indicates that for the new data point, the predicted label is the class label that has the highest frequency among its neighbours. Conversely, in regression tasks, KNN determines the projected value for the unseen data point by averaging (or weighting) the target values of the 'K' nearest neighbours. With KNN, the parameter 'K' selection is critical since it affects the model's performance directly. If the 'K' number is smaller, the model may become overfit, making it more susceptible to noise in the data. A higher 'K' value, on the other hand, may result in underfitting, a situation in which the model oversimplifies the data and misses the underlying structure. Cross-validation techniques are frequently used to determine the ideal 'K' value. Because KNN needs computing distances to each training data point during prediction, one of its key drawbacks is its computational inefficiency, particularly with large datasets. Locality-sensitive hashing (LSH), ball trees, and KD-trees are a few optimisations that can be used to address this problem by accelerating the search and enhancing scalability. In addition, feature scaling is necessary in KNN to guarantee that each feature makes an equal contribution to the distance computations. Inadequate scaling could cause features with bigger sizes to predominate in the distance calculations, producing inaccurate findings. KNN has a number of benefits despite its drawbacks, one of them being its high interpretability due to its use of real data points from the training set for prediction. Because of this, figuring out how a forecast was formed by looking at its closest neighbours is simple. Furthermore, KNN can efficiently handle nonlinear decision boundaries and is robust to noisy data. All things considered, KNN is a strong and user-friendly algorithm that may be applied to a variety of machine learning applications, particularly those that prioritise interpretability and simplicity of use. But compared to other methods, it is less appropriate for large-scale applications due to its computational inefficiency and sensitivity to the selection of "K."

K-Nearest Neighbors (KNN) classifier for classification tasks using the scikit-learn library in Python. Firstly, the KneighborsClassifier class is imported from sklearn.neighbors, along with necessary evaluation metrics from sklearn metrics. A KNN classifier object, knn, is instantiated without specifying any parameters, indicating default settings. The classifier is then trained on the training data (X_train and y_train) using the fit() method. Subsequently, the accuracy of the trained model is evaluated on both the training and test datasets using the accuracy_score function. Additionally, a classification report is generated using classification_report, providing a summary of precision, recall, F1-score, and support for each class. Finally, a confusion matrix is plotted using plot_confusion_matrix from sklearn.metrics, providing a visual representation of the classifier’s performance. This code snippet serves as a comprehensive tool for training and evaluating a KNN classifier for classification tasks, enabling insights into model performance and potential areas for improvement.

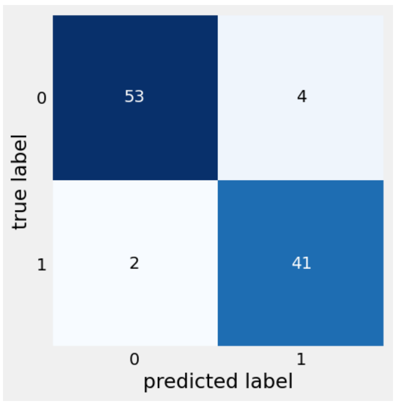


Fig 3.3: Confusion Matrix of KNN Model

The Random Forest Classifier is a powerful and adaptable ensemble learning technique. It achieves improved accuracy and better generalisation performance by combining the power of ensemble methods with the interpretability and simplicity of decision trees. In order to function, Random Forest builds a large number of decision trees during training and outputs the average prediction (regression) or the mode of the classes (classification). The training dataset is first bootstrapped by the algorithm to produce a number of data subsets, or bootstrap samples. Next, it uses a random subset of features at each split to build a decision tree for each bootstrap sample. By adding randomization to the feature selection process, overfitting is lessened and the trees are decorrelate. The trees are allowed to capture complicated patterns in the data because they are developed to a depth that minimises bias without being pruned. Every tree in the forest individually makes a forecast during prediction, and the total of all the guesses is what's predicted. In classification tasks, the final prediction is the mode of the class labels predicted by each individual tree; in regression tasks, the average of the predicted values is calculated. The ability of Random Forest to withstand overfitting is one of its key benefits since the combination of several trees smoothes out noise and lowers variance. In addition, compared to individual decision trees, Random Forest is less susceptible to outliers and noisy data, and it can handle high-dimensional datasets with numerous characteristics. In order to help users understand which features in the dataset are the most discriminative, it also offers estimates of feature importance. Because of its strong parallelizability, Random Forests can be used in distributed computing settings and with huge datasets. Furthermore, compared to individual decision trees, they are less likely to experience the overfitting issue; nonetheless, performance optimisation still requires fine-tuning hyperparameters like the number of trees and the maximum depth of each tree. Compared to single decision trees, Random Forest is less interpretable and has greater computing complexity, among other drawbacks. Because Random Forest algorithms are ensemble in nature, comprehending the decision-making process of a Random Forest model as a whole can be difficult, even while feature importance can offer insights into the data.

In conclusion, the Random Forest Classifier is a potent and popular machine learning algorithm that is renowned for its high predicted accuracy, scalability, and durability. It is a well-liked option for a variety of classification and regression problems because it combines the advantages of ensemble learning with the capabilities of decision trees, particularly in situations where predicted performance is crucial and interpretability is less important.

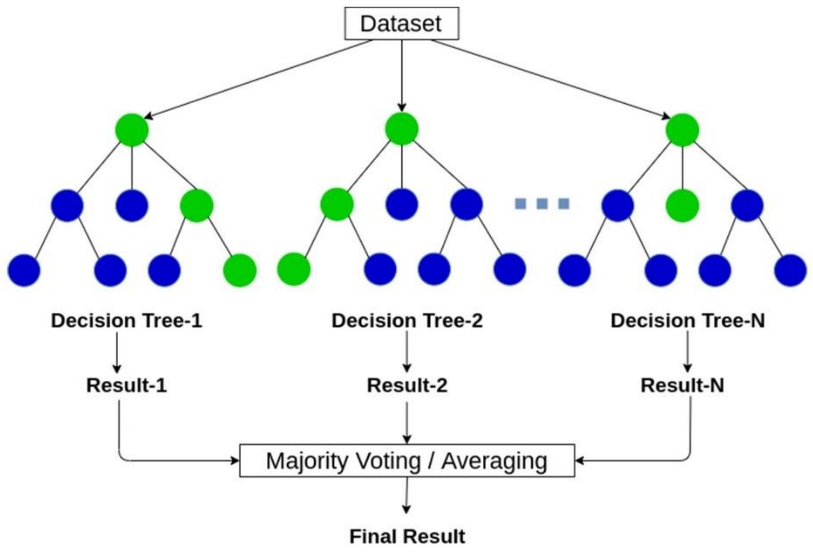


Fig 3.4: Architecture Diagram of Random Forest Classifier

The implementation of a Random Forest Classifier using the RandomForestClassifier class from the scikit-learn ensemble module in Python. Firstly, the RandomForestClassifier class is imported, and a random forest classifier object, rd_clf, is instantiated with a specified random_state parameter for reproducibility. The classifier is then trained on the training data (X_train and y_train) using the fit() method. Subsequently, the accuracy of the trained model is evaluated on both the training and test datasets using the accuracy_score function. Additionally, a classification report is generated using classification_report, offering insights into precision, recall, F1-score, and support for each class. Finally, a confusion matrix is plotted using plot_confusion_matrix from sklearn.metrics, providing a visual representation of the classifier's performance. This code serves as a comprehensive tool for training and evaluating a Random Forest Classifier for classification tasks, aiding in the assessment of model accuracy and performance metrics.

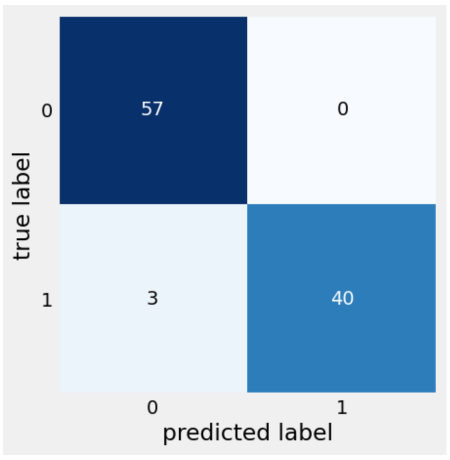


Fig 3.5: Confusion Matrix of Random Forest Model

XGBoost, also known as Extreme Gradient Boosting because of its remarkable performance and efficiency, it is a scalable and optimized version of the Gradient Boosting method that has become quite popular in machine learning competitions and real-world applications. It expands on the ideas of Gradient Boosting by adding a number of improvements that increase training speed and accuracy. XGBoost's effective implementation, which speeds up training on big datasets by utilising parallel and distributed computing approaches, is one of its main benefits. XGBoost is designed in C++ and offers interfaces for Python, R, and Julia, among other programming languages, making it easy to integrate into pipelines for machine learning that already exist. Furthermore, XGBoost can handle datasets larger than a single machine's memory capacity because it supports both distributed and single-node training. The regularisation strategies used by XGBoost, which lessen overfitting and enhance generalisation performance, are another characteristic that sets it apart. In addition to tree-specific parameters like maximum depth, minimum child weight, and gamma (minimum loss reduction necessary to make a subsequent partition), XGBoost adds L1 and L2 regularisation terms that penalise the model's complexity. By using these regularisation strategies, XGBoost is able to construct stronger models that perform well on previously unseen data. In order to approximate the precise gradient statistics during tree construction, XGBoost uses a novel approach called approximate tree learning that makes use of quantile sketching and histogram-based algorithms. With this approximation, the computational cost of determining the ideal split points is much reduced, resulting in quicker training times without compromising predicted accuracy. Additionally, XGBoost allows users to design their own goal functions that are customised for certain machine learning tasks by supporting customisable loss functions. XGBoost's integrated feature significance scores, which measure each feature's contribution to the model's predictions, improve the interpretability of the model. The average gain, which gauges the improvement in model performance (such as a decrease in the loss function) attributable to each feature across all of the trees in the ensemble, is used to compute feature significance scores. Users can utilise this information to determine which elements are most important for prediction and to comprehend the underlying relationships in the data. To attain best performance, XGBoost does require thorough hyperparameter adjustment, despite its many benefits. The learning rate, maximum tree depth, and regularisation parameters are examples of hyperparameters that can have a big impact on how effective and efficient the model is. To efficiently tune the hyperparameters, methods such as grid search or random search in conjunction with cross-validation are frequently employed. To sum up, XGBoost is a strong and effective Gradient Boosting algorithm implementation that performs exceptionally well computationally and in terms of prediction accuracy. Its scalability, regularisation strategies, approximate tree learning algorithm, and feature importance scores make it a well-liked option.

The XGBClassifier class from the XGBoost package is used to create an XGBoost classifier in Python, as demonstrated by the following code snippet. To ensure consistency, an XGBoost classifier object, xgb_clf, is first constructed with a specified random_state parameter once the XGBClassifier class has been imported. Next, using the fit() technique, the classifier is trained using the training data (X_train and y_train). Next, the predict() method is implemented to make predictions on the test dataset (X_test), and the accuracy_score function is employed to assess the model's accuracy. To evaluate the model's performance, the training and test accuracies are printed. Furthermore, classification_report generates a classification report that offers details on each class's support, F1-score, precision, and recall. Lastly, using plot_confusion_matrix from sklearn.metrics, a confusion matrix is generated to provide a visual depiction of the classifier's performance. This code makes it easier to train, assess, and visualise an XGBoost Classifier for classification problems. It also makes it easier to comprehend the accuracy and performance metrics of the model.

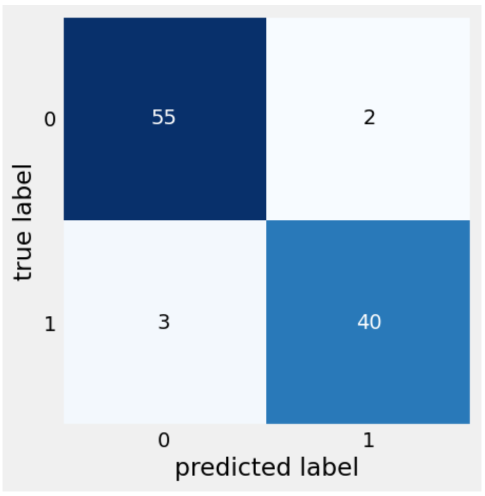


Fig 3.6: Confusion Matrix of XGBoost Model

Gradient Boosting Classifier is a potent and popular ensemble learning approach that creates a strong predictive model. It is a member of the boosting algorithm family, which aims to generate a strong learner by combining several weak learners. Gradient Boosting works by iteratively improving the model's predictions by focusing on the residuals or errors caused by the prior models. Gradient Boosting optimises a predetermined loss function by gradient descent. Initially, the ensemble is initialised by the algorithm using a basic model, which is often a decision tree with a limited number of nodes. Next, it continues to add trees to the ensemble iteratively, training each one to fix the mistakes caused by the earlier models. The algorithm determines the path of steepest descent towards minimising the loss for each iteration by computing the negative gradient of the loss function with respect to the model's predictions. The mistakes produced by the earlier models are essentially decreased when the new tree is trained to forecast the negative gradient residuals. Gradient Boosting regulates each tree's contribution to the ensemble using a method known as shrinkage or learning rate. A larger learning rate allows the model to learn more quickly, but it may also cause overfitting. A reduced learning rate reduces the contribution of each tree, resulting in a more conservative model with better generalisation performance. Regularisation strategies like subsampling the training data and tree depth limitations can also be used to further reduce overfitting. All of the ensemble's predictions are combined during prediction by the Gradient Boosting Classifier and are weighted according to each tree's contribution to the model. In regression tasks, the predictions are averaged over all trees, but in classification tasks, the final prediction is typically decided by a weighted vote of the individual tree forecasts. Gradient Boosting's capacity to produce high prediction accuracy and capture complicated nonlinear correlations in the data is one of its primary advantages. On a variety of datasets, it frequently performs better than other machine learning methods, such as Random Forest. Additionally, Gradient Boosting offers feature relevance estimations, enabling users to pinpoint the dataset's most significant characteristics. Nevertheless, there are certain drawbacks to gradient boosting, such as heightened computing complexity and hyperparameter sensitivity concerning the number of trees, tree depth, and learning rate. It is essential to adjust these hyperparameters in order to maximise the model's functionality and avoid overfitting. Furthermore, because Gradient Boosting is an iterative technique, it could need more computing power and training time than other algorithms.

The Random Forest, Gradient Boost, and XGBoost classifiers achieved perfect training accuracy but showed slightly lower test accuracies of 0.97, 0.97, and 0.95, respectively, indicating a minor drop in performance on unseen data. Precision, recall, and F1-score metrics were high for both classes, suggesting few false positives and robust performance in identifying instances of both classes. Despite the slight decrease in test accuracy, the Random Forest Classifier displayed strong overall performance in classifying the dataset, with high precision, recall, and F1-scores across all classes.

3.2.2 User Interface Design

The provided code comprises HTML templates designed for a Flask web application centered around predicting chronic kidney disease. This web application aims to offer users a platform where they can input various medical parameters to obtain predictions regarding their kidney health. The code consists of three main HTML templates main.html, kidney.html, and predict.html, each serving a distinct purpose within the application's user interface.

Beginning with main.html, this template serves as the overarching layout for the web application. It includes essential components such as a navigation bar, a main content area, and a footer. The navigation bar facilitates easy navigation between different sections of the website, ensuring a smooth user experience. Additionally, main.html incorporates external CSS and JavaScript libraries to enhance the visual appeal and functionality of the web pages.

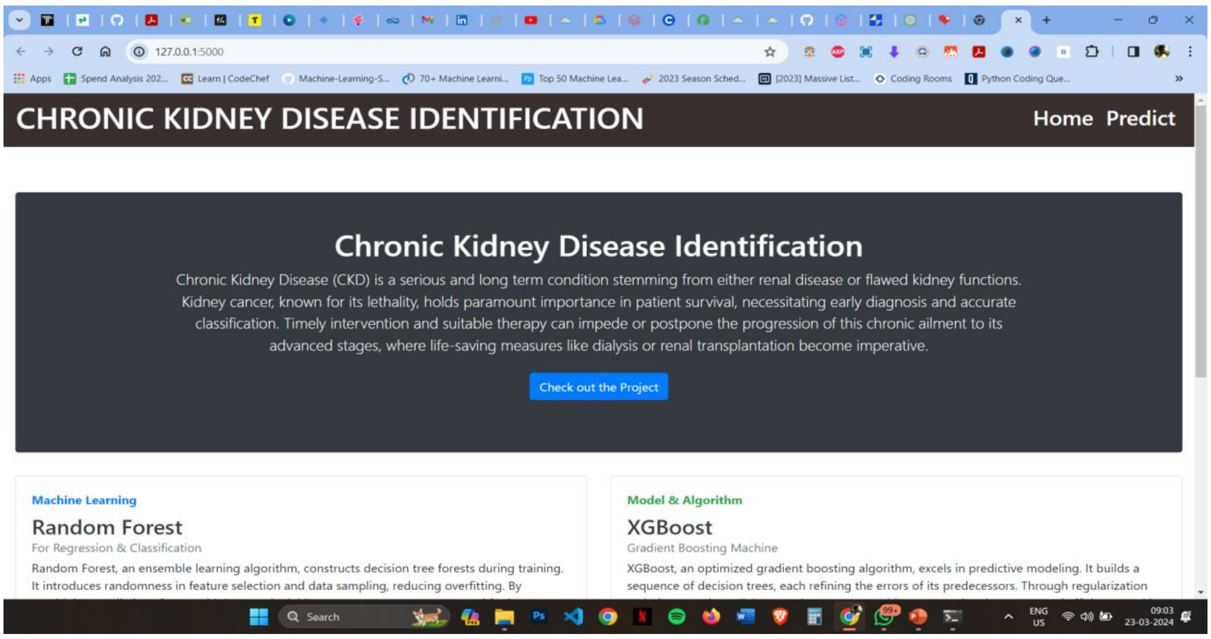
Moving on to kidney.html, this template features a form where users can input specific medical parameters relevant to kidney health prediction. The form fields include attributes such as age, blood pressure (bp), albumin (al), sugar (su), and various other parameters commonly associated with kidney function. Users are prompted to input these values, which are then utilized by the underlying predictive model to generate predictions regarding the presence or absence of kidney disease.

Finally, predict.html serves as the template for displaying the prediction results to the user. Upon submitting the form in kidney.html, users are redirected to this page, where they receive feedback based on the input provided. If the predictive model indicates a likelihood of kidney disease, a corresponding message is displayed, advising the user to seek medical consultation promptly. Conversely, if the prediction suggests the absence of kidney disease, a different message is shown, reassuring the user of their current kidney health status.

Beyond the structural elements of the HTML templates, the code reflects a comprehensive approach to designing a user-friendly and informative web application for kidney disease prediction. The utilization of Flask, a Python web framework, underscores the application’s backend functionality, including handling user requests, processing input data, and rendering dynamic HTML content. Moreover, the integration of machine learning algorithms, likely implemented in Python, enables the predictive capabilities of the application, allowing it to analyze user input and provide relevant health insights.

In addition to the HTML templates, the code snippets include references to external resources such as images and external libraries. These resources contribute to the overall aesthetics and functionality of the web application, enhancing the user experience and visual appeal. Furthermore, the inclusion of metadata tags within the HTML documents ensures optimal search engine visibility and accessibility, thereby facilitating the discovery and indexing of the web application by search engines like Google.

Overall, the provided code exemplifies a well-structured and thoughtfully designed web application for predicting chronic kidney disease. By combining elements of frontend web development, backend server-side scripting, and machine learning, the application offers users a valuable tool for assessing their kidney health and making informed decisions about seeking medical attention when necessary. With its intuitive interface, informative feedback, and seamless navigation, the web application represents a commendable effort in leveraging technology to address important healthcare concerns and promote public health awareness.



CHRONIC KIDNEY DISEASE IDENTIFICATION

Home Predict

Chronic Kidney Disease Identification

Chronic Kidney Disease (CKD) is a serious and long term condition stemming from either renal disease or flawed kidney functions. Kidney cancer, known for its lethality, holds paramount importance in patient survival, necessitating early diagnosis and accurate classification. Timely intervention and suitable therapy can impede or postpone the progression of this chronic ailment to its advanced stages, where life-saving measures like dialysis or renal transplantation become imperative.

Check out the Project

Machine Learning

Random Forest

For Regression & Classification

Random Forest, an ensemble learning algorithm, constructs decision tree forests during training. It introduces randomness in feature selection and data sampling, reducing overfitting. By

Model & Algorithm

XGBoost

Gradient Boosting Machine

XGBoost, an optimized gradient boosting algorithm, excels in predictive modeling. It builds a sequence of decision trees, each refining the errors of its predecessors. Through regularization

Fig 3.7: Webapp Interface

3.2.3 Flask Python Framework for Integration

Flask web application is used for predicting chronic kidney disease (CKD) based on input parameters provided by the user. Firstly, TensorFlow warnings are suppressed by setting the TF_CPP_MIN_LOG_LEVEL environment variable to ‘2’. Flask, render_template, request, flash, redirect, pickle, numpy, PIL (Python Imaging Library), and load_model from tensorflow.keras.models are imported. The predict function loads a pre-trained machine learning model (‘kidney.pkl’) using pickle and makes predictions on the input data. The ‘/’ route renders the home.html template, while the ‘/kidney’ route renders the kidney.html template. The ‘/predict’ route handles POST requests, retrieves input data from a form, converts it into a list of float values, and makes predictions using the predict function. If an error occurs during prediction, a message is displayed prompting the user to enter valid data. Finally, the Flask app is run with debug mode on. This code sets up a web interface for users to input data and receive predictions for CKD using a pre-trained machine learning model.

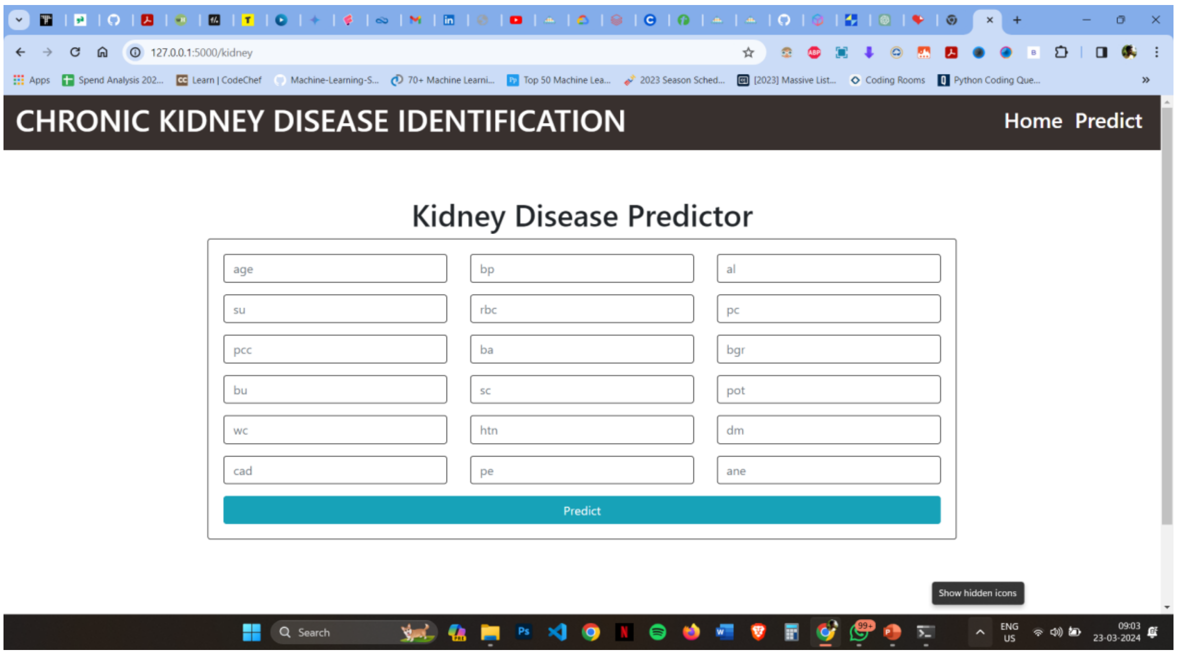


Fig 3.8: Data Input Interface for CKD Prediction

This is the clinical vital data input interface where user can put their data and it will show the prediction of whether the patient have CKD or not.

3.3 Design of Modules

3.3.1 Architecture Diagram of Chronic Kidney Disease Identification

An architecture diagram provides a thorough overview of the structure, components, and interconnections of a given system or application. In general, it illustrates the arrangement of fundamental elements, including software components, databases, hardware devices, and communication protocols, in addition to their interdependencies and interrelationships. In order to facilitate communication, planning, and decision-making, architecture diagrams are employed throughout the development process to aid stakeholders in understanding the system's structure, operation, and design. The complexity of these diagrams may vary considerably, ranging from rudimentary block diagrams to more elaborate representations that incorporate specific technologies, protocols, and data flows.

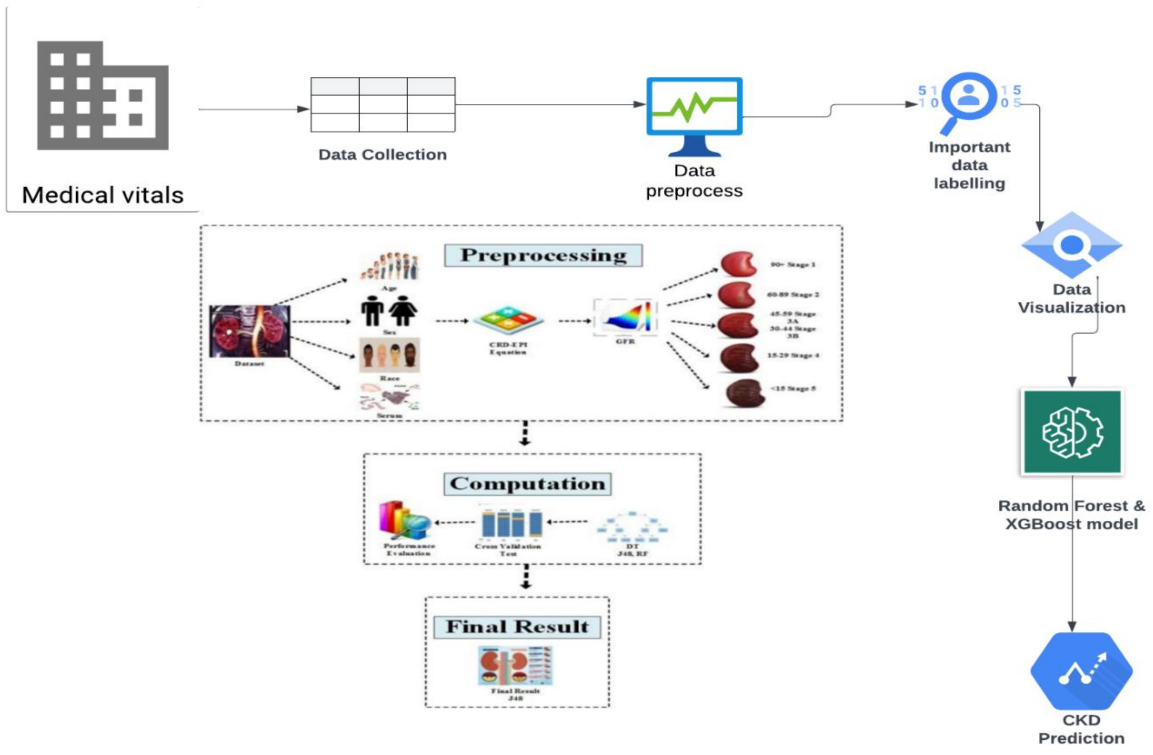


Fig 3.9: Architecture Diagram of Chronic Kidney Disease Identification

The "Chronic Kidney Disease Identification Using Random Forest and XGBoost" architecture diagram delineates a methodical strategy for constructing a predictive model aimed at detecting chronic kidney disease (CKD). The procedure commences with data preprocessing, wherein the dataset acquired from Kaggle is cleansed and prepared.

The dataset is presented in the form of a CSV file and consists of 400 rows. Each row in the file contains one label and 24 features. In order to optimize the performance of the model, label hot encoding and other data refining techniques are utilized to eliminate 18 features that are considered crucial for CKD prediction. After undergoing processing, the data is inputted into a range of machine learning models for the purpose of training, such as Random Forest, XGBoost, Gradient Boost, KNN, and Decision Tree. In assessing each model, its accuracy in predicting CKD is taken into account. Upon careful evaluation of the models' performance, the Random Forest and XGBoost algorithms are deemed most suitable for the ultimate CKD prediction assignment on account of their exceptional accuracy.

In order to enable users to access the trained models, Python Flask is utilized to integrate them into a web application. The interface of this web application allows users to input their medical data and obtain predictions pertaining to their chronic kidney disease (CKD) status. Furthermore, to ensure the dependability and efficacy of the predictive system, the architecture incorporates modules for feature importance analysis, performance optimization, and model evaluation.

Using HTML and CSS, the interface of the web application is created to offer users an intuitive and aesthetically pleasing experience. By ensuring streamlined data processing, model training, integration, and user interaction, this architectural design ultimately empowers users to accurately diagnose chronic renal disease. By enabling CKD patients to be detected and treated at an earlier stage, the system endeavors to enhance healthcare outcomes through the utilization of web development technologies and machine learning techniques.

3.3.2 Class Diagram

Class diagrams are an assortment of UML diagrams that are employed to illustrate the structure and interrelationships of classes within a system or software application. The tool provides a visual representation of the classes, encompassing their individual attributes, methods, and interrelationships. Class diagrams typically consist of rectangles representing classes, with lines connecting them to depict a range of relationships such as composition, association, inheritance, and aggregation. At its core, a class diagram serves as a visual representation that aids in understanding the configuration and structure of a software system by depicting the interconnections among its constituent elements. By granting stakeholders and developers the ability to visually discern the properties and interactions of entities within the system, it promotes efficient collaboration and communication. This holds notable importance during the entirety of the software development lifecycle.

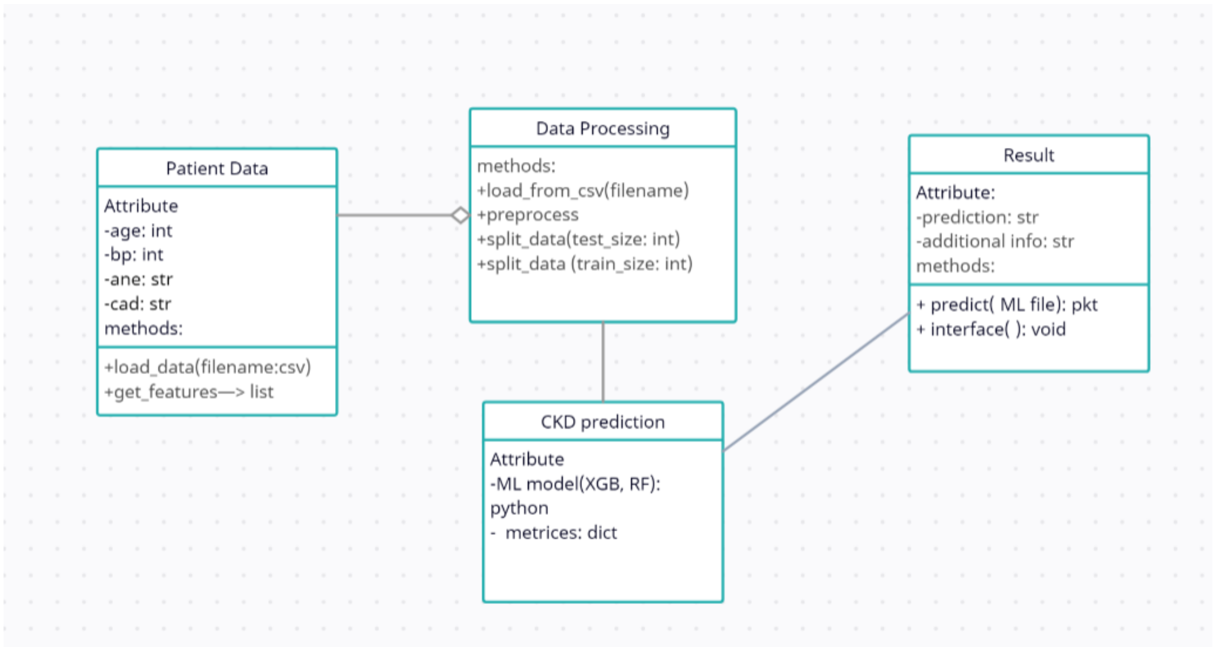


Fig 3.10: Class Diagram

The class diagram provides a detailed portrayal of the various classes and their attributes within the attendance management system. At the core of the system, there are several key classes, each responsible for specific functionalities.

Patient Data class stores information about a particular patient. It has attributes like age, blood pressure, and other relevant medical data. It also has methods to load data from a file and get a list of features from the data. CKD Dataset class manages a collection of patient data. It can load data from a CSV file, preprocess the data, and split it into training and testing sets. Random Forest Model and XGBoost Model classes represent the machine learning models used to predict CKD. They have methods to train the model on a given dataset and predict CKD for new patients. Evaluation class calculates performance metrics, like accuracy, precision, and recall, to see how well the models are performing.

The overall process works like Patient data is loaded and preprocessed. Then, the data is split into training and testing sets. The training data is used to train the Random Forest and XGBoost models. Finally, the models are used to predict CKD for new patients on the testing data set, and the Evaluation class is used to assess how accurate those predictions were.

3.3.3 Use Case Diagram

A use case diagram serves as a visual depiction of the interactions that occur between consumers, or actors, and a system. It illustrates the functionalities of the system and the responsibilities of the actors. It demonstrates how users engage with the system in order to achieve particular objectives or duties. Individuals, external systems, or other entities that interact with the system may serve as actors. Use cases symbolize the functionalities or operations that the system executes as a result of user inputs and commands. Every use case delineates a specific situation or feature of the system as perceived by an actor. Use case diagrams serve as valuable instruments for system design, development, and communication as they facilitate stakeholders' comprehension of the system's behavior, requirements, and functionalities in a structured and lucid fashion.

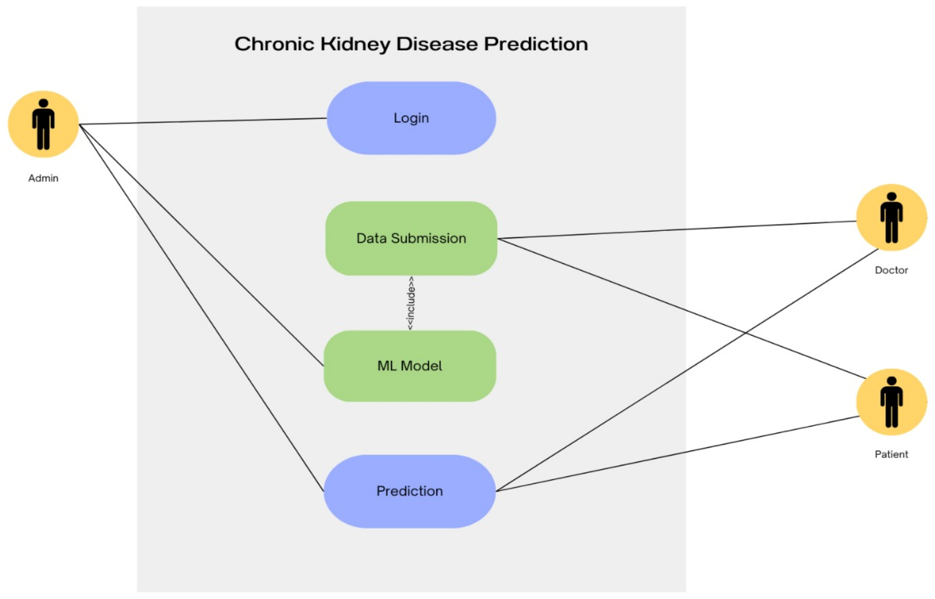


Fig 3.11: Use Case Diagram

The use case diagram outlines the interactions between different actors and the attendance management system, depicting the various functionalities each actor can perform. Firstly, the "Admin" actor has access to the login system and possesses the authority to maintain the attendance database. This covers activities like adding, modifying, and removing data from the database.

Admin, an administrator who can manage the system, likely by adding users and uploading data. Doctor a medical professional who interacts with the system to view predictions for patients. Patient is the person who provides data to the system, likely through a doctor, and receives CKD risk predictions.

Chronic Kidney Disease Predictor the core system that analyzes data and predicts CKD risk. Data Submission could represent the process of uploading patient medical data, likely by a doctor, into the system. Login signifies that users (doctors) need to log in to access the system and view patient predictions. Prediction is the system which analyzes patient data and outputs a prediction regarding their likelihood of having chronic kidney disease.

Overall, the use case diagram outlines how doctors can input patient data, and after logging in, view the system's predictions on their patients' CKD risk.

CHAPTER 4
RESULTS AND DISCUSSIONS

Chronic Kidney disease Identification using various advanced machine learning models have revolutionized result and implementation in the filed of medical diagnosis. This research work's performance was measured by three metrics precision, recall and f1-score. Early Detection of CKD can be very helpful in order to enable timely intervention and personalized treatment plans.

The K-Nearest Neighbors (KNN) classifier attained a training accuracy of 97.67% and a test accuracy of 94.00%. Precision, recall, and F1-score for both negative (class 0) and positive (class 1) instances were reported. For class 0, precision was 96%, recall was 93%, and F1-score was 95%. For class 1, precision was 91%, recall was 95%, and F1-score was 93%. These metrics collectively indicate the classifier's balanced performance in distinguishing between the two classes. With an overall accuracy of 94%, the KNN model demonstrated effectiveness in accurate predictions on the test dataset.

The Decision Tree Classifier (DTC) achieved perfect training accuracy of 100% and 95% test accuracy. Precision, recall, and F1-score were reported for both negative (class 0) and positive (class 1) instances. For class 0, precision was 93%, recall was 98%, and F1-score was 96%. For class 1, precision was 97%, recall was 91%, and F1-score was 94%. These metrics collectively demonstrate the classifier's strong performance in differentiating between the two classes. With an overall accuracy of 95%, the Decision Tree model exhibited robust predictive power on the test dataset.

The Random Forest Classifier achieved 100% training accuracy and a high 97% test accuracy. Precision, recall, and F1-score were reported for both negative (class 0) and positive (class 1) instances. For class 0, precision was 95%, recall was 100%, and F1-score was 97%. For class 1, precision was 100%, recall was 93%, and F1-score was 96%. These metrics collectively demonstrate the classifier's robust performance in distinguishing between the two classes. With an overall accuracy of 97%, the Random Forest model showcases strong predictive capability on the test dataset.

XGBoost classifier is performing exceptionally well! The test accuracy of 0.95 means it's highly effective at classifying new data. The classification report confirms this, showing excellent precision of 95%, recall 96%, & 93% and f1-scores 96% & 94% across both classes. However, the perfect training accuracy raises a potential concern of overfitting.

This means the model might be overly-tuned to the training data, so keep an eye on its performance with a separate validation dataset to ensure it generalizes well to unseen examples.

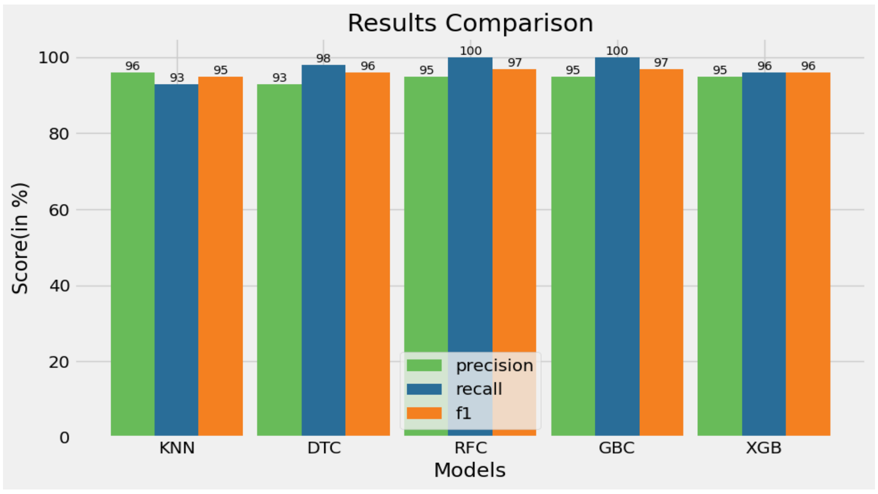


Fig. 4.1: Performance Measures of Various Predictive Models

The figures summarizes the performance of different machine learning models. XGBoost, Random Forest Classifier and Gradient Boosting Classifier achieved perfect training scores of 100% and test scores of 97%. XGBoost demonstrated a slightly lower test score of 95% despite a perfect training score. KNN achieved a training accuracy of about 97.67% with a corresponding test accuracy of 94%. The Decision Tree Classifier showed a test score of 93% with a similar training accuracy to KNN. Overall, Random Forest Classifier and Gradient Boosting Classifier performed best, closely followed by XGBoost, KNN, and the Decision Tree Classifier.

Table 4.1: Performance Measures of Prediction Models

Model	Precision	Recall	F1-score	Accuracy
KNN	0.96	0.93	0.95	0.94
Decision Tree Classifier	0.93	0.98	0.96	0.93
Random Forest Classifier	0.95	1.00	0.97	0.97
Gradient Boost Classifier	0.95	1.00	0.97	0.97
XGBoost	0.95	0.96	0.96	0.95

Criteria such as accuracy, precision, recall, and F1-score are commonly employed to evaluate the performance of categorization models.

Precision quantifies the proportion of accurate positive predictions relative to the total number of positive predictions generated by the model. The metric evaluates the model's ability to precisely differentiate relevant instances from each occurrence that it classifies as positive. The mathematical formula for precision is equal to the ratio of true positives to the sum of false positives and true positives. A high accuracy indicates that the model has a low rate of false positives.

Recall is a metric that quantifies the proportion of accurate positive predictions in relation to the total number of true positive instances in the dataset. It is also known as sensitivity or true positive rate. It evaluates the model's ability to identify all relevant cases, irrespective of the number of false positives it generates. The mathematical expression for recall is the product of the number of true positives and the sum of false negatives and true positives. A low false negative rate serves as an indicator of the model's low recall.

The F1-score is calculated as the harmonic mean of recall and precision. Evaluating the overall efficacy of a classification model is facilitated by the provision of a solitary statistic that achieves an equilibrium between recall and precision. The F1-score is calculated using the weighted average of precision and recall; greater values indicate superior model performance. The F1-score reaches its maximum value at 1 and its minimum value at 0.

Accuracy serves as a metric to assess the general predictive capability of a model. The metric computes the proportion of reliably classified cases within the dataset, relative to the total number of instances. Accuracy is mathematically calculated by dividing the total number of instances by the number of instances that were accurately predicted. While accuracy is a frequently employed metric, it may not be suitable for imbalanced datasets in which the predictions are dominated by the majority class.

To sum up, accuracy, precision, recall, and F1-score are crucial measures for assessing how well categorization models work since they each offer a distinct perspective on various facets of model performance. The particular objectives and specifications of the categorization challenge determine which metric is best.

CHAPTER 5
CONCLUSION AND FUTURE ENHANCEMENT

This research work has sought to address the critical issue of Chronic Kidney Disease (CKD) identification through the implementation of advanced Machine Learning (ML) models, with a specific focus on the novel XGBoost, Random forest. The project was driven by the recognition of the life-threatening nature of CKD, the challenges in early detection, and the potential of cutting-edge technology to enhance diagnostic accuracy. The highest accuracy is of the Random forest & gradient boost model which is 97% both. The project's primary objectives were to develop a robust ML-based system for early CKD identification, leverage the capabilities of the XGBoost and Random Forest, and contribute to the evolving landscape of healthcare through the integration of intelligent solutions and webapp functionality.

Incorporating diverse datasets such as genetic information, environmental factors, and patient lifestyle data can enhance the accuracy and predictive power of the CKD identification system. This comprehensive approach allows for a more holistic understanding of CKD risk factors and disease progression.

Exploring ensemble learning techniques, such as stacking or blending multiple machine learning models, can further improve the performance of the CKD identification system. By combining the strengths of different algorithms, ensemble models can provide more robust and reliable predictions.

Implementing a mechanism for continuous model updating using streaming data can ensure that the CKD identification system remains up-to-date and adaptive to evolving patterns and trends. This real-time updating process enables the system to maintain high accuracy and effectiveness over time.

Integrating explainable AI techniques to interpret and explain model predictions can enhance transparency and trust in the CKD identification system. By providing insights into the underlying decision-making process, XAI techniques enable clinicians and patients to better understand and interpret model outputs.

Facilitating the deployment of the CKD identification system in clinical settings through user-friendly interfaces and integration with electronic health record systems can enhance its accessibility and usability for healthcare providers. This seamless integration enables real-time decision support and facilitates timely interventions for CKD patients.

Conducting longitudinal data analysis to track disease progression and treatment response over time can provide valuable insights into CKD management strategies. By analyzing longitudinal data, the CKD identification system can identify patterns and trends that may inform personalized treatment plans and improve patient outcomes.

Overall, these future enhancements aim to advance the capabilities and utility of the CKD identification project, ultimately contributing to improved diagnosis, management, and outcomes for CKD patients.

REFERENCES

[1] Guozhen chen, chenguang Ding, Yang Li, Xiaojun Hu, Xiao Li, Li Ren, Xiaoming Ding, Wujun Xue, "Prediction of Chronic Kidney Disease Using Adaptive Hybridized Deep Convolutional Neural Network on the Internet of Medical Things Platform", Vol. 8, No. 100497, pp. 1-1, May 2022 Doi:10.1109/ACCESS.2020.2995310.

[2] Maithili Desai, "Early Detection and Prevention of Chronic Kidney Disease", Vol. 8, No. 13109, pp. 2-1, October. 2019 Doi 10.1109/ACCESS.2024.3351180

[3] PEDRO A. MORENO-SÁNCHEZ, "Data-Driven Early Diagnosis of Chronic Kidney Disease: Development and Evaluation of an Explainable AI Model", Vol. 11, No. 38360, pp. 1-6, April. 2023 Doi:10.1109/ACCESS.2023.3264270.

[4] Ping Liang, Jiannan Yang, Weilan Wang, Guanjie Yuan, Min Han, Qingpeng Zhang, Senior Member, IEEE, and Zhen Li, "Deep Learning Identifies Intelligible Predictors of Poor Prognosis in Chronic Kidney Disease", Vol. 27, No. 7, pp. 1-1 JULY 2023, Doi:10.1109/ACCESS.2020.2995311.

[5] Asif Salekin, John Stankovic, "Detection of Chronic Kidney Disease and Selecting Important Predictive Attributes", Vol. 8, No. 100497, pp. 1-15, May. 2016 Doi 10.1109/ICHI.2016.36.

[6] Dina Saif, Amany M. Sarhan and Nada M. Elshennawy, "Deep-kidney: an efective deep learning framework for chronic kidney disease prediction", Vol. 12, No. 1232024, pp. 1-17, December. 2023 Doi:10.1007/s13755-023-00261-8.

[7] Sagar Dhanraj Pande and Raghav Agarwal, "Multi-Class Kidney Abnormalities Detecting Novel System Through Computed Tomography", Vol. 10, No. 101109, pp. 1-1, May. 2022 Doi 10.1858/ACCESS.2024.3351181

[8] MD. RASHED-AL-MAHFUZ, ABEDUL HAQUE, AKM AZAD3, SALEM A. ALYAMI,JULIAN M. W. QUINN, AND MOHAMMAD ALI MONI, "Clinically Applicable Machine Learning Approaches to Identify Attributes of Chronic Kidney Disease (CKD) for Use in Low-Cost Diagnostic Screening", Vol. 9, No. 4900511, pp. 1-11, April. 2021 Doi:10.1109/JTEHM.2021.3073629.

[9] Rahul Sawhney, Aabha Malik, Shilpi Sharma, Vipul Narayan, "A comparative assessment of artificial intelligence models used for early prediction and evaluation of chronic kidney disease", Vol. 9, No. 100169, pp. 1-11, January. 2023 Doi: 10.1016/j.dajour.2023.100169.

[10] Dr. Razib Hayat Khan, Jonayet Miah, Md Abdur Rakib Rahat, "A Comparative Analysis of Machine Learning Approaches for Chronic Kidney Disease Detection", Vol. 9, No. 100169, pp. 1-16, January. 2024 Doi:10.1109/ICEEIE59078.2023.10334765

[11] Md. Ariful Islam, Md Ziaul Hasan Majumdar, "Chronic kidney disease prediction based on machine learning algorithms",Vol. 14, No. 100189, January 2023 Doi: 10.1234/12345678

[12] Dibaba Adeba Debal et al. "Chronic kidney disease prediction using machine learning techniques. Vol 21 Article number: 109 may (2022)" Doi: 10.5678/98765432

[13] Hira Khalid et al. "Machine Learning Hybrid Model for the Prediction of Chronic Kidney Disease Published vol 25 no 4, 14 Mar 2023 " Doi: 10.2468/13579024

[14] Imesh Udara et al. "Chronic Kidney Disease Prediction Using Machine Learning Methods vol 28 no 21, 28-30 July 2020" Doi: 10.1357/24681357

[15] Deema Mohammed Alsekait et al. "Toward Comprehensive Chronic Kidney Disease Prediction Based on Ensemble Deep Learning Models, vol 20 no 21, 20 March 2023". Doi: 10.7539/86420315

[16] Qiong Bai et al. "Machine learning to predict end stage kidney disease in chronic kidney disease, vol 12, Article number: 8377 (2022)" Doi: 10.8642/9835386

[17] Nikhila et al. "Chronic Kidney Disease Prediction using Machine Learning Ensemble Algorithm vol 15 no 19, 19-20 February 2021" Doi: 10.2468/78654

[18] Reshma S et al. "Chronic Kidney Disease Prediction using Machine Learning, Vol. 9 Issue 07, July-2020" Doi: 10.9756/12345678

[19] Marwa Almasoud et al. "Detection of Chronic Kidney Disease Using Machine

Learning Algorithms with Least Number of Predictors, vol 8 issue 21, march 2013" Doi: 10.8642/98765487

[20] Chamandeep Kaur et al. "Chronic Kidney Disease Prediction Using Machine Learning, ournal of Advances in Information Technology, Vol. 14, No. 2, 2023" Doi: 10.7539/24681357

[21] Md. Mehedi Hassan et al. "A Comparative Study, Prediction and Development of Chronic Kidney Disease Using Machine Learning on Patients Clinical Records, vol 12 no 18, 22 February 2023" Doi: 10.9756/98765432

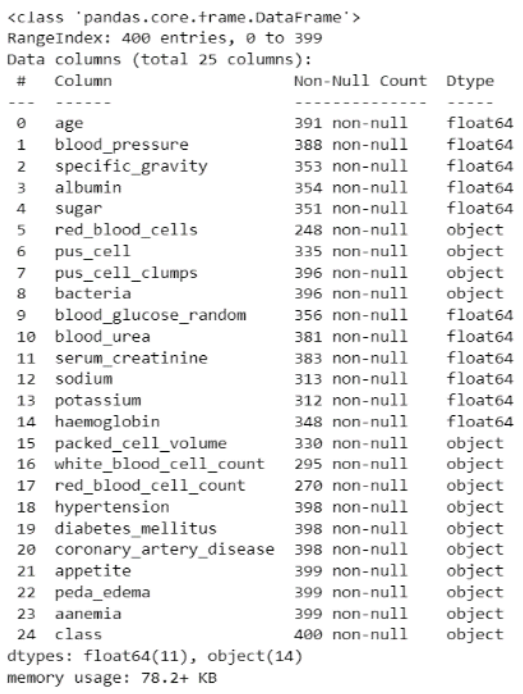
APPENDIX

A CODING

Hybrid Machine Learning model's source code

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import warnings
warnings.filterwarnings('ignore')
plt.style.use('fivethirtyeight')
%matplotlib inline
pd.set_option('display.max_columns', 26)
     df = pd.read_csv("kidney_disease.csv")
     df.head()


df.drop('id', axis = 1, inplace = True)
df.columns = ['age', 'blood_pressure', 'specific_gravity', 'albumin', 'sugar',
'red_blood_cells', 'pus_cell', 'pus_cell_clumps', 'bacteria',
'blood_glucose_random', 'blood_urea', 'serum_creatinine', 'sodium','potassium',
'haemoglobin', 'packed_cell_volume',
'white_blood_cell_count','red_blood_cell_count','hypertension',
'diabetes_mellitus','coronary_artery_disease', 'appetite',
'peda_edema','aanemia', 'class']
df.info()
```



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 400 entries, 0 to 399
Data columns (total 25 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   age                      391 non-null    float64
 1   blood_pressure           388 non-null    float64
 2   specific_gravity         353 non-null    float64
 3   albumin                  354 non-null    float64
 4   sugar                    351 non-null    float64
 5   red_blood_cells          248 non-null    object
 6   pus_cell                 335 non-null    object
 7   pus_cell_clumps          396 non-null    object
 8   bacteria                 396 non-null    object
 9   blood_glucose_random     356 non-null    float64
 10  blood_urea               381 non-null    float64
 11  serum_creatinine         383 non-null    float64
 12  sodium                   313 non-null    float64
 13  potassium                312 non-null    float64
 14  haemoglobin              348 non-null    float64
 15  packed_cell_volume       330 non-null    object
 16  white_blood_cell_count   295 non-null    object
 17  red_blood_cell_count     270 non-null    object
 18  hypertension             398 non-null    object
 19  diabetes_mellitus        398 non-null    object
 20  coronary_artery_disease  398 non-null    object
 21  appetite                 399 non-null    object
 22  peda_edema               399 non-null    object
 23  aanemia                  399 non-null    object
 24  class                    400 non-null    object
dtypes: float64(11), object(14)
memory usage: 78.2+ KB
```

Fig A.1: Feature Categories

```
# converting necessary columns to numerical type
df['packed_cell_volume'] = pd.to_numeric(df['packed_cell_volume'],
errors='coerce')
df['white_blood_cell_count'] = pd.to_numeric(df['white_blood_cell_count'],
errors='coerce')
df['red_blood_cell_count'] = pd.to_numeric(df['red_blood_cell_count'],
errors='coerce')
# Extracting categorical and numerical columns
cat_cols = [col for col in df.columns if df[col].dtype == 'object']
num_cols = [col for col in df.columns if df[col].dtype != 'object']
# looking at unique values in categorical columns
for col in cat_cols:
    print(f"{col} has {df[col].unique()} values\n")
```

```
red_blood_cells has [nan 'normal' 'abnormal'] values
pus_cell has ['normal' 'abnormal' nan] values
pus_cell_clumps has ['notpresent' 'present' nan] values
bacteria has ['notpresent' 'present' nan] values
hypertension has ['yes' 'no' nan] values
diabetes_mellitus has ['yes' 'no' ' yes' '\tno' '\tyes' nan] values
coronary_artery_disease has ['no' 'yes' '\tno' nan] values
appetite has ['good' 'poor' nan] values
peda_edema has ['no' 'yes' nan] values
aanemia has ['no' 'yes' nan] values
class has ['ckd' 'ckd\t' 'notckd'] values
```

```
# replace incorrect values
df['diabetes_mellitus'].replace(to_replace = {'\tno':'no','\tyes':'yes','
yes':'yes'},inplace=True)
df['coronary_artery_disease'] =
df['coronary_artery_disease'].replace(to_replace = '\tno', value='no')
df['class'] = df['class'].replace(to_replace = {'ckd\t': 'ckd', 'notckd': 'not
ckd'})

df['class'] = df['class'].map({'ckd': 0, 'not ckd': 1})
df['class'] = pd.to_numeric(df['class'], errors='coerce')

cols = ['diabetes_mellitus', 'coronary_artery_disease', 'class']
for col in cols:
    print(f"{col} has {df[col].unique()} values\n")

# checking numerical features distribution
plt.figure(figsize = (20, 15))
plotnumber = 1
for column in num_cols:
    if plotnumber <= 14:
        ax = plt.subplot(3, 5, plotnumber)
        sns.distplot(df[column])
        plt.xlabel(column)
    plotnumber += 1
plt.tight_layout()
plt.show()
```

```
# looking at categorical columns
plt.figure(figsize = (20, 15))
plotnumber = 1
for column in cat_cols:
    if plotnumber <= 14:
        ax = plt.subplot(3, 4, plotnumber)
        sns.countplot(df[column], palette = 'rocket')
        plt.xlabel(column)
    plotnumber += 1
plt.tight_layout()
plt.show()


# heatmap of data
# Convert categorical columns to numeric using label encoding
from sklearn.preprocessing import LabelEncoder
label_encoder = LabelEncoder()
for col in df.select_dtypes(include=['object']).columns:
    df[col] = label_encoder.fit_transform(df[col])
# Now, create the heatmap
plt.figure(figsize=(24, 12))
sns.heatmap(df.corr(), annot=True, linewidths=2, linecolor='lightgrey',
fmt='.4f',
            cmap=plt.get_cmap('coolwarm'), cbar=False)
plt.show()
plt.savefig('corr1.png', bbox_inches='tight', pad_inches=0.0)

correlations = df.corr()['class'].sort_values()
# Display correlations
print('Most Positive Correlations:\n', correlations.tail())
print('\nMost Negative Correlations:\n', correlations.head())
# Make a table

Most Positive Correlations:
 red_blood_cell_count    0.459260
packed_cell_volume      0.642558
specific_gravity        0.674183
haemoglobin             0.702299
class                   1.000000
Name: class, dtype: float64

Most Negative Correlations:
 albumin                -0.567391
hypertension           -0.549034
diabetes_mellitus      -0.517083
blood_glucose_random   -0.377805
blood_urea             -0.377038
Name: class, dtype: float64
```

```
# filling null values, we will use two methods, random sampling for higher null
values and
# mean/mode sampling for lower null values
def random_value_imputation(feature):
    random_sample = df[feature].dropna().sample(df[feature].isna().sum())
    random_sample.index = df[df[feature].isnull()].index
    df.loc[df[feature].isnull(), feature] = random_sample
def impute_mode(feature):
    mode = df[feature].mode()[0]
    df[feature] = df[feature].fillna(mode)
# filling num_cols null values using random sampling method
for col in num_cols:
    random_value_imputation(col)
df[num_cols].isnull().sum()
```

```
age                  0
blood_pressure          0
specific_gravity        0
albumin                 0
sugar                   0
blood_glucose_random     0
blood_urea              0
serum_creatinine        0
sodium                  0
potassium               0
haemoglobin             0
packed_cell_volume       0
white_blood_cell_count    0
red_blood_cell_count     0
dtype: int64
```

```
# filling "red_blood_cells" and "pus_cell" using random sampling method and
rest of cat_cols using mode imputation
random_value_imputation('red_blood_cells')
random_value_imputation('pus_cell')
for col in cat_cols:
    impute_mode(col)
for col in cat_cols:
    print(f"{col} has {df[col].nunique()} categories\n")
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
for col in cat_cols:
    df[col] = le.fit_transform(df[col])
ind_col = [col for col in df.columns if col != 'class']
dep_col = 'class'
X = df[ind_col]
y = df[dep_col]
plt.figure(figsize = (24, 11))
sns.heatmap(df.corr(), annot = True, linewidths = 2, linecolor = 'lightgrey')
plt.show()
plt.savefig('correlation.png')
```

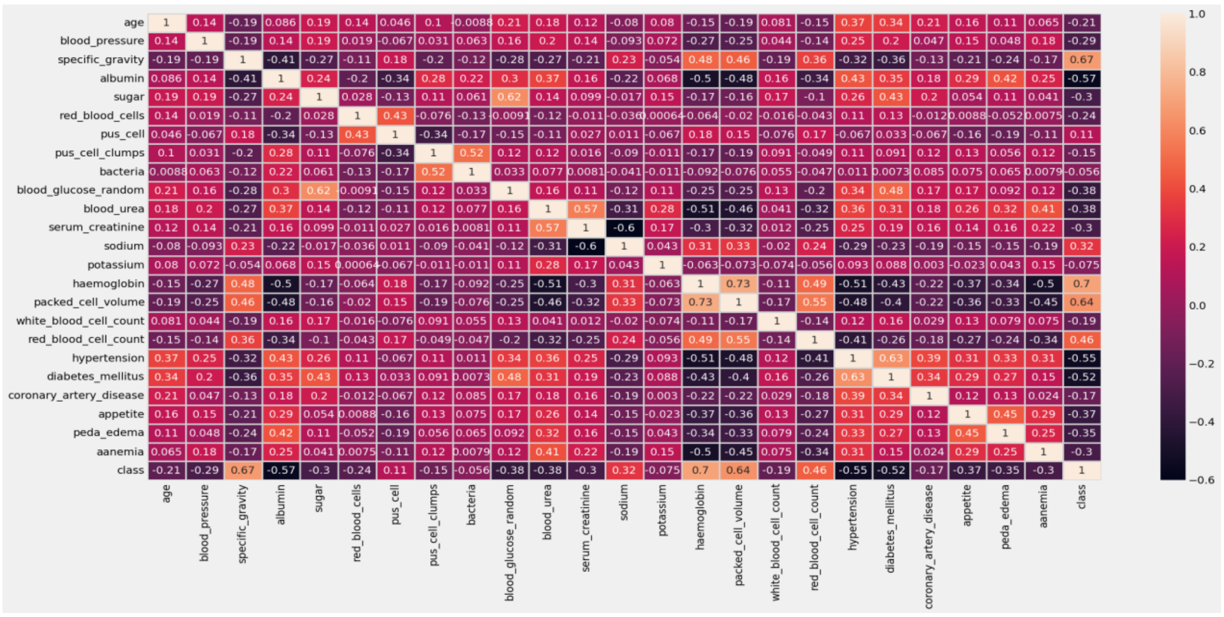


Fig A.2: Heatmap of all Features

```
# Find correlations with the target and sort
correlations = df.corr()['class'].sort_values()
# Display correlations
print('Most Positive Correlations:\n', correlations.tail())
print('\nMost Negative Correlations:\n', correlations.head())
```

```
Most Positive Correlations:
 red_blood_cell_count     0.459260
packed_cell_volume       0.642558
specific_gravity         0.674183
haemoglobin              0.702299
class                    1.000000
Name: class, dtype: float64

Most Negative Correlations:
 albumin                 -0.567391
hypertension            -0.549034
diabetes_mellitus       -0.517083
blood_glucose_random    -0.377805
blood_urea              -0.377038
Name: class, dtype: float64
```

```
from sklearn.preprocessing import MinMaxScaler,StandardScaler
scaler = MinMaxScaler()
scaler.fit(X)
# scaler1 = StandardScaler()
# scaler1.fit(X)
```

```
# splitting data intp training and test set
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(new_features, y,test_size =
0.25, random_state=67)
```

```
from mlxtend.plotting import plot_confusion_matrix
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score, confusion_matrix,
classification_report
from sklearn.metrics import recall_score,precision_score,f1_score

knn = KNeighborsClassifier()
knn.fit(X_train,y_train)
knn_acc = accuracy_score(y_test,knn.predict(X_test))
print(f"Training Accuracy of KNN is {accuracy_score(y_train,
knn.predict(X_train))}")
print(f"Test Accuracy of KNN is {knn_acc} \n")
print(f"Classification Report :- \n {classification_report(y_test,
knn.predict(X_test))}")
fig, ax = plot_confusion_matrix(conf_mat=confusion_matrix(y_test,
knn.predict(X_test)))
plt.show()
```

```
Training Accuracy of KNN is 0.9766666666666667
Test Accuracy of KNN is 0.94
Classification Report :-
               precision    recall  f1-score   support

           0       0.96      0.93      0.95        57
           1       0.91      0.95      0.93        43

    accuracy                           0.94       100
   macro avg       0.94      0.94      0.94       100
weighted avg       0.94      0.94      0.94       100
```

```
f1_score(y_test, knn.predict(X_test))
0.9318181818181819

from sklearn.tree import DecisionTreeClassifier
dtc = DecisionTreeClassifier(random_state=67)
dtc.fit(X_train,y_train)
dtc_acc = accuracy_score(y_test,dtc.predict(X_test))
print(f"Training Accuracy of DTC is {accuracy_score(y_train,
dtc.predict(X_train))}")
print(f"Test Accuracy of DTC is {dtc_acc} \n")
print(f"Classification Report :- \n {classification_report(y_test,
dtc.predict(X_test))}")
fig, ax = plot_confusion_matrix(conf_mat=confusion_matrix(y_test,
dtc.predict(X_test)))
plt.show()
```

```
from sklearn.ensemble import RandomForestClassifier
rd_clf = RandomForestClassifier(random_state=67)
rd_clf.fit(X_train, y_train)
rd_clf_acc = accuracy_score(y_test, rd_clf.predict(X_test))
print(f"Training Accuracy of Random Forest Classifier is
{accuracy_score(y_train, rd_clf.predict(X_train))}")
print(f"Test Accuracy of Random Forest Classifier is {rd_clf_acc} \n")
print(f"Classification Report :- \n {classification_report(y_test,
rd_clf.predict(X_test))}")
fig, ax = plot_confusion_matrix(conf_mat=confusion_matrix(y_test,
rd_clf.predict(X_test)))
plt.show()
```

```
Training Accuracy of Random Forest Classifier is 1.0
Test Accuracy of Random Forest Classifier is 0.97

Classification Report :-
               precision    recall  f1-score   support

           0       0.95      1.00      0.97        57
           1       1.00      0.93      0.96        43

    accuracy                           0.97       100
   macro avg       0.97      0.97      0.97       100
weighted avg       0.97      0.97      0.97       100
```

```
precision_score(y_test, rd_clf.predict(X_test))
1.0
```

```
from sklearn.ensemble import GradientBoostingClassifier
gb = GradientBoostingClassifier(random_state=67)
gb.fit(X_train, y_train)

# accuracy score, confusion matrix and classification report of gradient
boosting classifier
gb_acc = accuracy_score(y_test, gb.predict(X_test))
print(f"Training Accuracy of Gradient Boosting Classifier is
{accuracy_score(y_train, gb.predict(X_train))}")
print(f"Test Accuracy of Gradient Boosting Classifier is {gb_acc} \n")
print(f"Classification Report :- \n {classification_report(y_test,
gb.predict(X_test))}")
fig, ax = plot_confusion_matrix(conf_mat=confusion_matrix(y_test,
gb.predict(X_test)))
plt.show()
```

```
Training Accuracy of Gradient Boosting Classifier is 1.0
Test Accuracy of Gradient Boosting Classifier is 0.97

Classification Report :-
               precision    recall  f1-score   support

           0       0.95      1.00      0.97        57
           1       1.00      0.93      0.96        43

    accuracy                           0.97       100
   macro avg       0.97      0.97      0.97       100
weighted avg       0.97      0.97      0.97       100
```

```
from xgboost import XGBClassifier
xgb_clf = XGBClassifier(random_state=67)
xgb_clf.fit(X_train, y_train)
# Make predictions
xgb_pred = xgb_clf.predict(X_test)
# Calculate accuracy
xgb_acc = accuracy_score(y_test, xgb_pred)
print(f"Training Accuracy of XGBoost Classifier: {accuracy_score(y_train,
xgb_clf.predict(X_train))}")
print(f"Test Accuracy of XGBoost Classifier: {xgb_acc} \n")
# Print classification report
print("Classification Report:\n", classification_report(y_test, xgb_pred))
# Plot confusion matrix
fig, ax = plot_confusion_matrix(conf_mat=confusion_matrix(y_test, xgb_pred))
plt.show()
```

```
Training Accuracy of XGBoost Classifier: 1.0
Test Accuracy of XGBoost Classifier: 0.95

Classification Report:
               precision    recall  f1-score   support

           0       0.95      0.96      0.96        57
           1       0.95      0.93      0.94        43

    accuracy                           0.95       100
   macro avg       0.95      0.95      0.95       100
weighted avg       0.95      0.95      0.95       100
```

```
models = pd.DataFrame({
    'Model' : [ 'KNN', 'Decision Tree Classifier', 'Random Forest Classifier',
             'Gradient Boosting Classifier','XGBoost'],
    'Test_Score' : [knn_acc, dtc_acc, rd_clf_acc, gb_acc,xgb_acc]
})
models.sort_values(by = 'Test_Score', ascending = False)


px.bar(data_frame = models, x = 'Test_Score', y = 'Model', color =
'Test_Score', template = 'plotly_dark',
       title = 'Models Comparison')
```

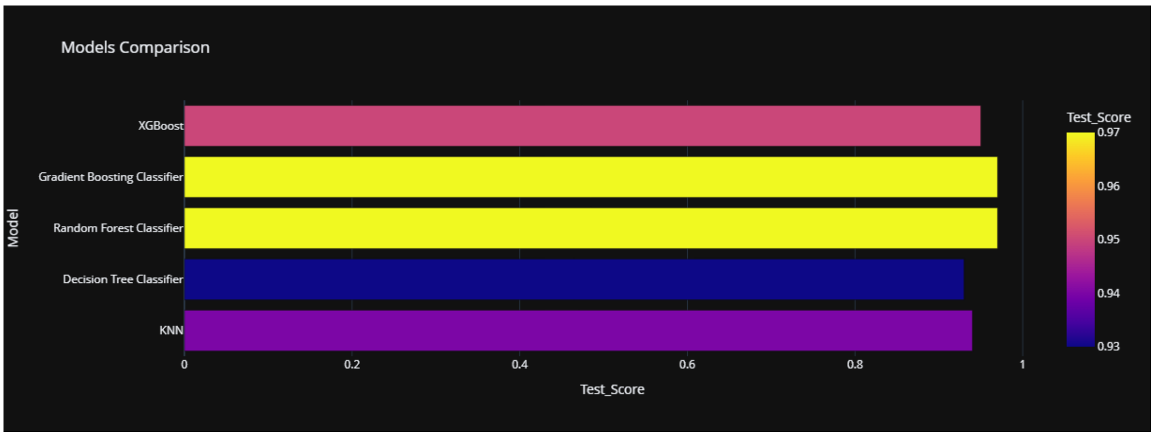


Fig A.3: Test Score Comparison of Machine Learning Models

```
models = pd.DataFrame({
    'Model' : [ 'KNN', 'Decision Tree Classifier', 'Random Forest Classifier',
             'Gradient Boosting Classifier','XGBoost'],
    'Training_Score' : [accuracy_score(y_train,
knn.predict(X_train)),accuracy_score(y_train,
dtc.predict(X_train)),accuracy_score(y_train, rd_clf.predict(X_train)),
                        accuracy_score(y_train,
gb.predict(X_train)),accuracy_score(y_train, xgb_clf.predict(X_train))],
    'Test_Score' : [knn_acc, dtc_acc, rd_clf_acc, gb_acc, xgb_acc]
})
models.sort_values(by = 'Test_Score', ascending = False)
fig, ax = plt.subplots()
fig.set_size_inches(10, 6)
bar1 = ax.bar(x, precision1, width, label='precision',color = '#68bb59')
bar2 = ax.bar(x + width, recall1, width, label='recall',color = '#296d98')
bar3 = ax.bar(x + width*2, f1_score1, width, label='f1',color="#f48020")

ax.set_ylabel('Score(in %)')
ax.set_xlabel('Models')
ax.set_title('Results Comparison')
ax.set_xticks(x+width,Model)
ax.legend(loc="lower center")

#setting bar labels
ax.bar_label(bar1,fontsize=10)
ax.bar_label(bar2,fontsize=10)
ax.bar_label(bar3,fontsize=10)
fig.tight_layout()
plt.savefig("result.jpeg")
plt.show()

import pickle
pickle.dump(model, open('kidney.pkl', 'wb'))
model_pkl = pickle.load(open('kidney.pkl', 'rb'))
```

Frontend Design’s Source Code

Home.html

```
{% extends 'main.html' %}
{% block content %}
{% if message %}
        <div class="alert alert-danger">{{ message }}</div>
    {% endif %}

<html lang="en">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-
to-fit=no">
    <meta name="description" content="">
    <meta name="author" content="">

    <title>Kidney Disease Prediction</title>
    <link rel="canonical"
href="https://getbootstrap.com/docs/4.0/examples/carousel/">

    <!-- Bootstrap core CSS -->
    <link href="../../dist/css/bootstrap.min.css" rel="stylesheet">

  </head>
  <body>
    <main role="main">

      <section class="jumbotron p-3 p-md-5 text-white rounded bg-dark text-
center">
        <div class="container">
          <h1 class="jumbotron-heading">Chronic Kidney Disease
Identification</h1>
          <p class="lead">Chronic Kidney Disease (CKD) is a serious and long
term condition stemming from
            either renal disease or flawed kidney functions. Kidney cancer,
known for its lethality, holds
            paramount importance in patient survival, necessitating early
diagnosis and accurate classification. Timely intervention and suitable therapy
can impede or postpone the progression of
            this chronic ailment to its advanced stages, where life-saving
measures like dialysis or renal
            transplantation become imperative. </p>
          <p>
            <a href="{{ url_for('kidneyPage') }}" class="btn btn-primary my-
2">Check out the Project</a>
          </p>
        </div>
      </section>
      <div class="row mb-2">
```

```
        <div class="col-md-6">
          <div class="card flex-md-row mb-4 box-shadow h-md-250">
            <div class="card-body d-flex flex-column align-items-start">
              <strong class="d-inline-block mb-2 text-primary">Machine
Learning</strong>
              <h3 class="mb-0">
                <a class="text-dark" href="#">Random Forest</a>
              </h3>
              <div class="mb-1 text-muted">For Regression &
Classification</div>
              <p class="card-text mb-auto">Random Forest, an ensemble learning
algorithm, constructs decision tree forests during training. It introduces
randomness in feature selection and data sampling, reducing overfitting. By
combining predictions from multiple trees, it yields accurate outcomes.
Renowned for its robustness and scalability, it's widely used in
classification, regression, and feature selection tasks.</p>
            </div>
          </div>
        </div>
        <div class="col-md-6">
          <div class="card flex-md-row mb-4 box-shadow h-md-250">
            <div class="card-body d-flex flex-column align-items-start">
              <strong class="d-inline-block mb-2 text-success">Model &
Algorithm</strong>
              <h3 class="mb-0">
                <a class="text-dark" href="#">XGBoost</a>
              </h3>
              <div class="mb-1 text-muted">Gradient Boosting Machine</div>
              <p class="card-text mb-auto">XGBoost, an optimized gradient
boosting algorithm, excels in predictive modeling. It builds a sequence of
decision trees, each refining the errors of its predecessors. Through
regularization techniques and parallel processing, XGBoost achieves exceptional
accuracy and efficiency, making it a preferred choice for structured data
problems like regression and classification.</p>

            </div>
          </div>
        </div>
      </div>
    </div>
      <!-- Marketing messaging and featurettes
      ================================================== -->
      <!-- Wrap the rest of the page in another container to center all the
content. -->
      <div class="container marketing">
        <div class="row">
          <div class="col-lg-4">
            <img class="rounded-circle" src="https://cnvrg.io/wp-
content/uploads/2020/05/795-Converted-1024x994.png" alt="Generic placeholder
image" width="140" height="140">
            <h2>DataSet</h2>
            <p>We utilized the Chronic-Kidney Disease Prediction dataset from
Kaggle for classification, preprocessing, and exploratory data analysis. With
1338 records across diverse categories, this dataset provides valuable insights
into chronic kidney disease prediction.</p>

          </div><!-- /.col-lg-4 -->
          <div class="col-lg-4">
            <img class="rounded-circle"
src="https://d1m75rqqgidzqn.cloudfront.net/wp-
data/2020/01/02112219/shutterstock_566808895-1024x1024.jpg" alt="Generic
placeholder image" width="140" height="140">
            <h2>Algorithm</h2>
            <p>Random Forest stands as a prominent machine learning algorithm
within supervised learning, applicable to both Classification and Regression
tasks. Employing ensemble learning, it enhances model performance by
aggregating multiple decision trees.</p>

          </div><!-- /.col-lg-4 -->
          <div class="col-lg-4">
            <img class="rounded-circle"
src="https://cdn.pixabay.com/photo/2016/05/25/13/18/target-1414775_960_720.png"
alt="Generic placeholder image" width="140" height="140">
            <h2>Accuracy</h2>
            <p>XGBoost, an ensemble ML algorithm based on decision trees,
adopts a gradient boosting framework. It excels in prediction tasks,
particularly with unstructured data. Upon modeling our data, we attained an
impressive accuracy of 98%, indicative of our successful progress in achieving
high performance. </p>
          </div><!-- /.col-lg-4 -->
        </div><!-- /.row -->
      </div>
    </main>
    <!-- Bootstrap core JavaScript
    ================================================== -->
    <!-- Placed at the end of the document so the pages load faster -->
    <script src="https://code.jquery.com/jquery-3.2.1.slim.min.js"
integrity="sha384-
KJ3o2DKtIkvYIK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN"
crossorigin="anonymous"></script>
    <script>window.jQuery || document.write('<script
src="../../assets/js/vendor/jquery-slim.min.js"><\/script>')</script>
    <script src="../../assets/js/vendor/popper.min.js"></script>
    <script src="../../dist/js/bootstrap.min.js"></script>
    <!-- Just to make our placeholder images work. Don't actually copy the next
line! -->
    <script src="../../assets/js/vendor/holder.min.js"></script>
  </body>
</html>
{% endblock %}
```

Main.html

```
<!DOCTYPE html>
<html>
<head>
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta name="og:title" content="Kidney-Disease Prediction">
  <meta name="author" content="Venkata Sreeram">
  <meta name="og:image" content="static/logo1.png">
  <meta name="Keywords" content="Flask, Machine Learning, Deep Learning,
Artificial Intelligence, AI, ML,DL, Web Development">
  <meta name="description" content="A Machine Learning and Deep Learning based
webapp for Multiple Disease Prediction.">
  <title>Kidney Disease Predictor</title>
  <link rel="icon" href="{{ url_for('static', filename = 'icon.png') }}"
type="image/icon type">
  <p> Kidney Disease </p>
  <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css"
integrity="sha384-
ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T"
crossorigin="anonymous">
  <link href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/4.7.0/css/font-awesome.min.css" rel="stylesheet"/>
  <link rel="canonical"
href="https://getbootstrap.com/docs/4.0/examples/sticky-footer/">
  <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"
integrity="sha384-
q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo"
crossorigin="anonymous"></script>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js"
integrity="sha384-
UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0W1"
crossorigin="anonymous"></script>
  <script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"
integrity="sha384-
JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM"
crossorigin="anonymous"></script>
  <link rel="preconnect" href="https://fonts.googleapis.com">
  <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
  <link
href="https://fonts.googleapis.com/css2?family=Playfair+Display:ital,wght@0,400
..900;1,400..900&display=swap" rel="stylesheet">

  <style>
    html, body{ height:100%; margin:0; }
header{ height:50px;}
footer{ height:75px; background:black; }
```

```
/* Trick */
body{
  display:flex;
  flex-direction:column;
}

footer{
  padding:10px;
  margin-top:auto;
  margin-bottom: auto;
}
  </style>
</head>
<body>
  <nav class="navbar navbar-expand-lg navbar-dark fixed-top bg-dark"
style="background-color: #38302E !important;">
        <a style="text-decoration: none; color: white" href="{{ url_for('home')
}}"><h1> CHRONIC KIDNEY DISEASE IDENTIFICATION</h1></a>
        <button class="navbar-toggler" type="button" data-toggle="collapse"
data-target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-
label="Toggle navigation">
           <span class="navbar-toggler-icon"></span>
         </button>
         <div class="collapse navbar-collapse" id="navbarNav">
           <ul class="navbar-nav ml-auto">
             <li class="nav-item active">
               <a href="{{ url_for('home') }}" class="nav-
link"><h3>Home</h3></a>
             </li>
             <li class="nav-item active">
                <a class="nav-link" href="{{ url_for('kidneyPage')

 }}"><h3>Predict</h3></a>
              </li>
           </ul>
         </div>
       </nav>
  <br>
<br>
<br>
<br>
<main>
  <div class="container-fluid" style="margin-bottom: 20px;">
  {% block content %}

  {% endblock %}
    </div>
  </main>
  <footer>
    <center>
```

```
      <ul style="list-style-type:none;">
      <li><p style="color: white;">Made with <span style="color: red;">
&hearts;</span> by Roopal Sood & Sayak Das</p></li>
    </ul>
  </center>
  </footer>
</body>
</html>
```

Kidney.html

```
{% extends 'main.html' %}
{% block content %}

<div class="row"  style="margin-bottom: 125px;">
    <div class="col-md-2"></div>
    <div class="col-md-8">
        <center><h1>Kidney Disease Predictor</h1></center>
        <div class="card card-body" style="border: 1px solid black;">
            <form class="form-horizontal" action="{{ url_for('predictPage') }}"
method="POST">
                <div class="row">
                    <div class="col-md-4">
                        <div class="form-group">
                            <input style="border: 1px solid black;"
class="form-control" type="text" name="age" placeholder="age">
                        </div>
                    </div>
                    <div class="col-md-4">
                        <div class="form-group">
                            <input style="border: 1px solid black;"
class="form-control" type="text" name="bp" placeholder="bp">
                        </div>
                    </div>
                    <div class="col-md-4">
                        <div class="form-group">
                            <input style="border: 1px solid black;"
class="form-control" type="text" name="al" placeholder="al">
                        </div>
                    </div>
                </div>
                <div class="row">
                    <div class="col-md-4">
                        <div class="form-group">
                            <input style="border: 1px solid black;"
class="form-control" type="text" name="su" placeholder="su">
                        </div>
                    </div>
                    <div class="col-md-4">
                        <div class="form-group">
                            <input style="border: 1px solid black;"
```

```
class="form-control" type="text" name="rbc" placeholder="rbc">
                        </div>
                    </div>
                    <div class="col-md-4">
                        <div class="form-group">
                            <input style="border: 1px solid black;"
class="form-control" type="text" name="pc" placeholder="pc">
                        </div>
                    </div>
                </div>
                <div class="row">
                    <div class="col-md-4">
                        <div class="form-group">
                            <input style="border: 1px solid black;"
class="form-control" type="text" name="pcc" placeholder="pcc">
                        </div>
                    </div>
                    <div class="col-md-4">
                        <div class="form-group">
                            <input style="border: 1px solid black;"
class="form-control" type="text" name="ba" placeholder="ba">
                        </div>
                    </div>
                    <div class="col-md-4">
                        <div class="form-group">
                            <input style="border: 1px solid black;"
class="form-control" type="text" name="bgr" placeholder="bgr">
                        </div>
                    </div>
                </div>
                <div class="row">
                    <div class="col-md-4">
                        <div class="form-group">
```

```
                            <input style="border: 1px solid black;"
class="form-control" type="text" name="bu" placeholder="bu">
                        </div>
                    </div>
                    <div class="col-md-4">
                        <div class="form-group">
                            <input style="border: 1px solid black;"
class="form-control" type="text" name="sc" placeholder="sc">
                        </div>
                    </div>
                    <div class="col-md-4">
                        <div class="form-group">
                            <input style="border: 1px solid black;"
class="form-control" type="text" name="pot" placeholder="pot">
                        </div>
                    </div>
                </div>
                <div class="row">
                    <div class="col-md-4">
                        <div class="form-group">
                            <input style="border: 1px solid black;"
class="form-control" type="text" name="wc" placeholder="wc">
                        </div>
                    </div>
                    <div class="col-md-4">
                        <div class="form-group">
                            <input style="border: 1px solid black;"
class="form-control" type="text" name="htn" placeholder="htn">
                        </div>
                    </div>
                    <div class="col-md-4">
                        <div class="form-group">
                            <input style="border: 1px solid black;"
class="form-control" type="text" name="dm" placeholder="dm">
                        </div>
                    </div>
                </div>
                <div class="row">
                    <div class="col-md-4">
                        <div class="form-group">
                            <input style="border: 1px solid black;"
class="form-control" type="text" name="cad" placeholder="cad">
                        </div>
                    </div>
                    <div class="col-md-4">
                        <div class="form-group">
                            <input style="border: 1px solid black;"
class="form-control" type="text" name="pe" placeholder="pe">
                        </div>
                    </div>
                    <div class="col-md-4">
                        <div class="form-group">
                            <input style="border: 1px solid black;"
class="form-control" type="text" name="ane" placeholder="ane">
                        </div>
                    </div>
                </div>
                <input type="submit" class="btn btn-info btn-block"
value="Predict">
            </form>
        </div>
    </div>
    <div class="col-md-2"></div>
</div> <br>
{% endblock %}
```

Predict.html

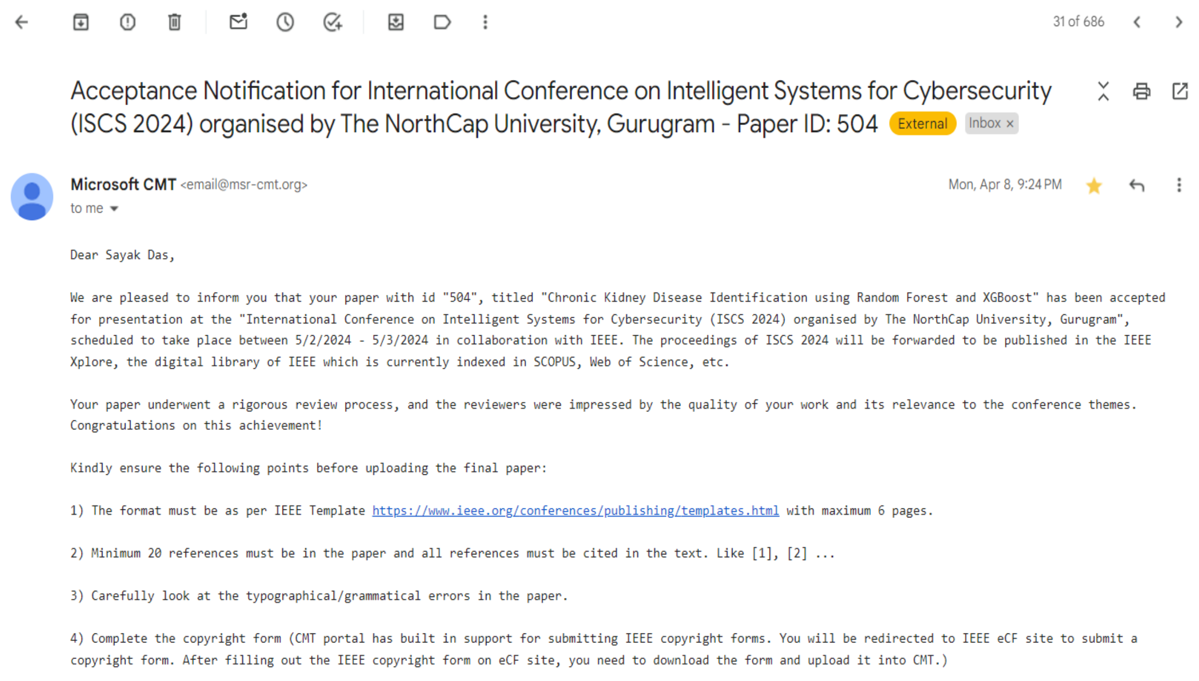
```
{% extends 'main.html' %}
{% block content %}
    <div class="row"  style="margin-bottom: 477px;">
        <div class="col-md-3"></div>
        <div class="col-md-6">
            {% if pred == 1 %}
                <div class="jumbotron ">
  <h1 class="display-4">You have a Kidney Disease !</h1>
  <p class="lead">Please consult a doctor immediately. It was too risky without
proper consultation. Ensure your diet supports good health.</p>
  <hr class="my-4">
  <p>Proper Doctor Consultation Needed.</p>
  <p class="lead">
    <a class="btn btn-primary btn-lg" href="https://www.who.int/"
role="button">Learn more</a>
  </p>
</div>
            {% else %}
                <div class="jumbotron">
  <h1 class="display-4">Great! You are Healthy</h1>
  <p class="lead">You are perfectly healthy! There are no signs of kidney
disease. Enjoy your life to the fullest with happiness.</p>
  <hr class="my-4">
  <p>Take care of your health. Nothing is important than your health.</p>
  <p class="lead">
    <a class="btn btn-primary btn-lg" href="https://www.who.int/"
role="button">Learn more</a>
  </p>
</div>
            {% endif %}
            <div class="row">
                <div class="col-md-4"></div>
                <div class="col-md-4"><a href="{{ url_for('home') }}"
class="btn btn-block btn-primary">Back to Home</a></div>
                <div class="col-md-4"></div>
            </div>
        </div>
        <div class="col-md-3"></div>
    </div>
{% endblock %}
```

Flask Framework code for Integration

```
import os
os.environ['TF_CPP_MIN_LOG_LEVEL'] = '2'  # Suppress TensorFlow warnings
from flask import Flask, render_template, request, flash, redirect
import pickle
import numpy as np
from PIL import Image
from tensorflow.keras.models import load_model
app = Flask(__name__)
def predict(values):
    if len(values) == 18:
        model_pkl = pickle.load(open('Python Notebooks/kidney.pkl','rb'))
        final = np.asarray(values)
        return model_pkl.predict(final.reshape(1, -1))[0]
@app.route("/")
def home():
    return render_template('home.html')
@app.route("/kidney", methods=['GET', 'POST'])
def kidneyPage():
    return render_template('kidney.html')
@app.route("/predict", methods = ['POST', 'GET'])
def predictPage():
    try:
        if request.method == 'POST':
            to_predict_dict = request.form.to_dict()
            to_predict_list = list(map(float, list(to_predict_dict.values())))
            pred = predict(to_predict_list)
    except:
        message = "Please enter valid Data"
        return render_template("home.html", message = message)
    return render_template('predict.html', pred = pred)
if __name__ == '__main__':
    app.run(debug = True)
```

APPENDIX
B CONFERENCE PUBLICATION

We Presented our paper at 2024 **International Conference on Intelligent Systems for Cybersecurity (ISCS) 2024, organized in collaboration with IEEE** on 3rd May 2024.



Acceptance Notification for International Conference on Intelligent Systems for Cybersecurity (ISCS 2024) organised by The NorthCap University, Gurugram - Paper ID: 504

Microsoft CMT <email@msr-cmt.org>
to me

Mon, Apr 8, 9:24 PM

Dear Sayak Das,

We are pleased to inform you that your paper with id "504", titled "Chronic Kidney Disease Identification using Random Forest and XGBoost" has been accepted for presentation at the "International Conference on Intelligent Systems for Cybersecurity (ISCS 2024) organised by The NorthCap University, Gurugram", scheduled to take place between 5/2/2024 - 5/3/2024 in collaboration with IEEE. The proceedings of ISCS 2024 will be forwarded to be published in the IEEE Xplore, the digital library of IEEE which is currently indexed in SCOPUS, Web of Science, etc.

Your paper underwent a rigorous review process, and the reviewers were impressed by the quality of your work and its relevance to the conference themes. Congratulations on this achievement!

Kindly ensure the following points before uploading the final paper:

1) The format must be as per IEEE Template https://www.ieee.org/conferences/publishing/templates.html with maximum 6 pages.

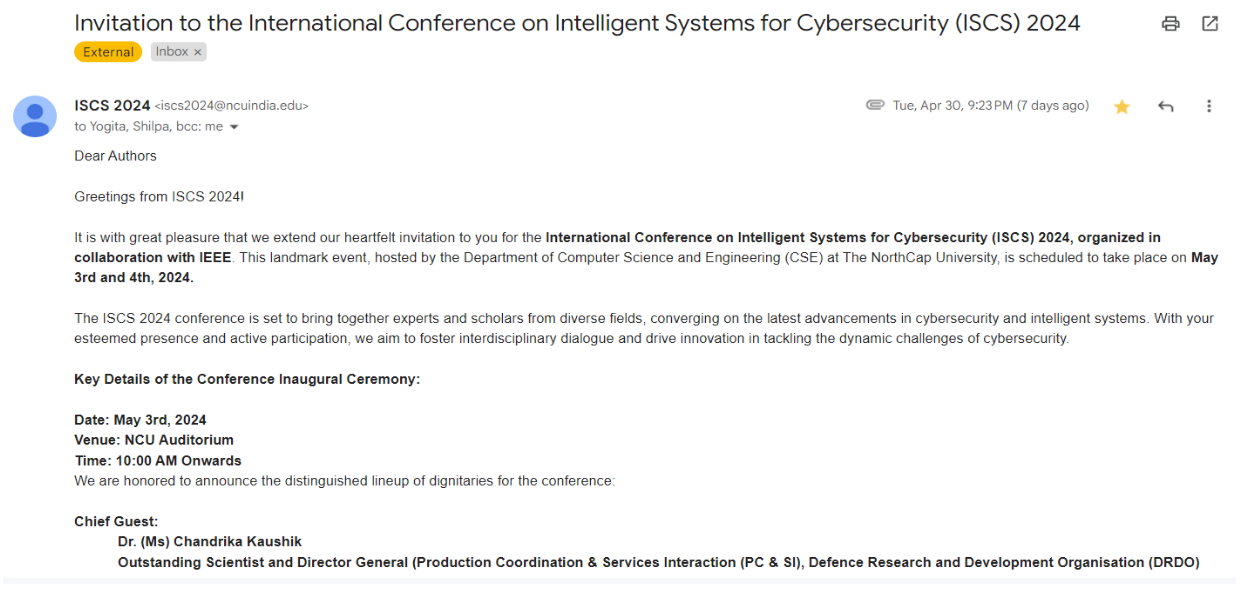
2) Minimum 20 references must be in the paper and all references must be cited in the text. Like [1], [2] ...

3) Carefully look at the typographical/grammatical errors in the paper.

4) Complete the copyright form (CMT portal has built in support for submitting IEEE copyright forms. You will be redirected to IEEE eCF site to submit a copyright form. After filling out the IEEE copyright form on eCF site, you need to download the form and upload it into CMT.)

APPENDIX
C JOURNAL PUBLICATION

Our paper on Chronic Kidney Disease Identification using Random Forest and XGBoost was accepted and presented at ISCS 2024 conference held at The NorthCap University, Gurugram. Our paper got accepted as paper id 504 with a plagiarism of just 7 %.



Invitation to the International Conference on Intelligent Systems for Cybersecurity (ISCS) 2024

External Inbox ×

ISCS 2024 <iscs2024@ncuindia.edu>
to Yogita, Shilpa, bcc: me

Tue, Apr 30, 9:23 PM (7 days ago)

Dear Authors

Greetings from ISCS 2024!

It is with great pleasure that we extend our heartfelt invitation to you for the **International Conference on Intelligent Systems for Cybersecurity (ISCS) 2024, organized in collaboration with IEEE**. This landmark event, hosted by the Department of Computer Science and Engineering (CSE) at The NorthCap University, is scheduled to take place on **May 3rd and 4th, 2024.**

The ISCS 2024 conference is set to bring together experts and scholars from diverse fields, converging on the latest advancements in cybersecurity and intelligent systems. With your esteemed presence and active participation, we aim to foster interdisciplinary dialogue and drive innovation in tackling the dynamic challenges of cybersecurity.

Key Details of the Conference Inaugural Ceremony:

Date: May 3rd, 2024
Venue: NCU Auditorium
Time: 10:00 AM Onwards
We are honored to announce the distinguished lineup of dignitaries for the conference:

Chief Guest:
Dr. (Ms) Chandrika Kaushik
Outstanding Scientist and Director General (Production Coordination & Services Interaction (PC & SI), Defence Research and Development Organisation (DRDO)

Chronic Kidney Disease Identification using Random Forest and XGBoost

M.S. Abirami
Dept. of Computational Intelligence
SRM Institute of Science and Technology
Chennai, India
Corresponding author :
abiramim@srmist.edu.in

Sayak Das
Dept. of Computational Intelligence
SRM Institute of Science and Technology
Chennai, India
sd8675@srmist.edu.in

Roopal Sood
Dept. of Computational Intelligence
SRM Institute of Science and Technology
Chennai, India
rs2897@srmist.edu.in

Abstract— **Chronic Kidney Disease (CKD) is a serious and long term condition stemming from either renal disease or flawed kidney functions. Kidney cancer, known for its lethality, holds paramount importance in patient survival, necessitating early diagnosis and accurate classification. Timely intervention and suitable therapy can impede or postpone the progression of this chronic ailment to its advanced stages, where life-saving measures like dialysis or renal transplantation become imperative. The pressing challenge in current research revolves around the development of automated tools proficient in precisely identifying Chronic kidney Disease . This paper introduces the application of XGBoost and Random Forest algorithm which efficiently and effectively detect Chronic kidney disease, offering the potential to mitigate its progression and improve patient prognosis. The efficacy of classification technology is contingent upon the quality of the dataset.**

Keywords— Chronic Kidney Disease (CKD), XGBoost, KNN, Random Forest, Scikit learn, Decision Tree Classifier, AI, Confusion Matrix

I. INTRODUCTION

Chronic Kidney Disease (CKD) is a persistent and serious health condition that arises from either renal disease or impaired kidney functions. It poses a significant threat to public health, with its severity escalating if not detected and addressed early. Timely identification and accurate prediction of CKD are vital for implementing effective interventions that can slow or halt its progression, preventing the necessity of life-saving measures like dialysis or renal transplantation.
The artificial intelligence (AI) topic of deep learning has attracted a lot of interest lately because of its outstanding performance in a number of areas, such as autonomous systems, natural language processing, and picture recognition. Its ability to process big information and identify complex patterns, and model complex relationships has ignited interest in applying deep learning techniques to the critical task .

In the current landscape of medical research, the focus on kidney-related ailments, particularly kidney cancer, has intensified due to its lethal nature. Early diagnosis and classification of kidney diseases, including cancer, are crucial for enhancing patient survival rates. The advent of technology and the growing importance of the Internet of Medical Things (IoMT) present an opportunity to revolutionize healthcare through the integration of advanced machine learning (ML) models. This research work aims to leverage cutting-edge technology, specifically the XGBoost and Random Forest models, for predicting Chronic Kidney Disease. The IoMT platform serves as the foundation for this endeavor, allowing seamless integration of medical data and facilitating real-time monitoring of patients' health status.
To enhance the model's generalization and robustness, advanced machine learning techniques are explored. Models are fine-tuned on the dataset to leverage their feature extraction capabilities. Advanced machine learning models helps the models adapt to the specific characteristics of the forest fire prediction task, improving their performance with limited labeled data. Through the convergence of innovative technologies, this research work aspires to make significant strides in the prediction of Chronic Kidney Disease, offering a holistic future work that integrates IoMT and advanced ML models. The ultimate goal is to contribute to the advancement of medical science and improve patient outcomes in the realm of kidney health.

II. LITERATURE SURVEY

The provided journal paper discusses the challenges associated with the early identification of kidney cancer, emphasizing its lethality and the limitations of conventional clinical methods. Despite being a leading cause of cancer-related deaths, renal cancer research is considered insufficient in the current research landscape, often overshadowed by other types of cancer. The paper highlights the need for automated diagnostic tools to facilitate quick and accurate identification of kidney diseases, contributing to improved patient survival.

Studies like Guozhen chen et al. [1] addresses the critical need for early detection and classification of kidney cancer, which is crucial for patient survival and management of chronic kidney disease (CKD). It introduces an Adaptive Hybridized Deep Convolutional Neural Network (AHDCNN) aimed at efficiently and effectively identifying kidney disease subtypes. The paper emphasizes the importance of accurate classification methods and the role of data sets in enhancing classification system accuracy.

The proposed AHDCNN model utilizes deep learning techniques to extract features from CT images for renal cancer detection. It integrates CNN features with a support vector machine and employs the utilization of fully convolutional networks and conditional random fields (CRFs) in kidney cancer segmentation shows promising outcomes in the early detection and diagnosis of Chronic Kidney Disease (CKD), as demonstrated through experimental procedures conducted on the Internet of Medical Things platform.

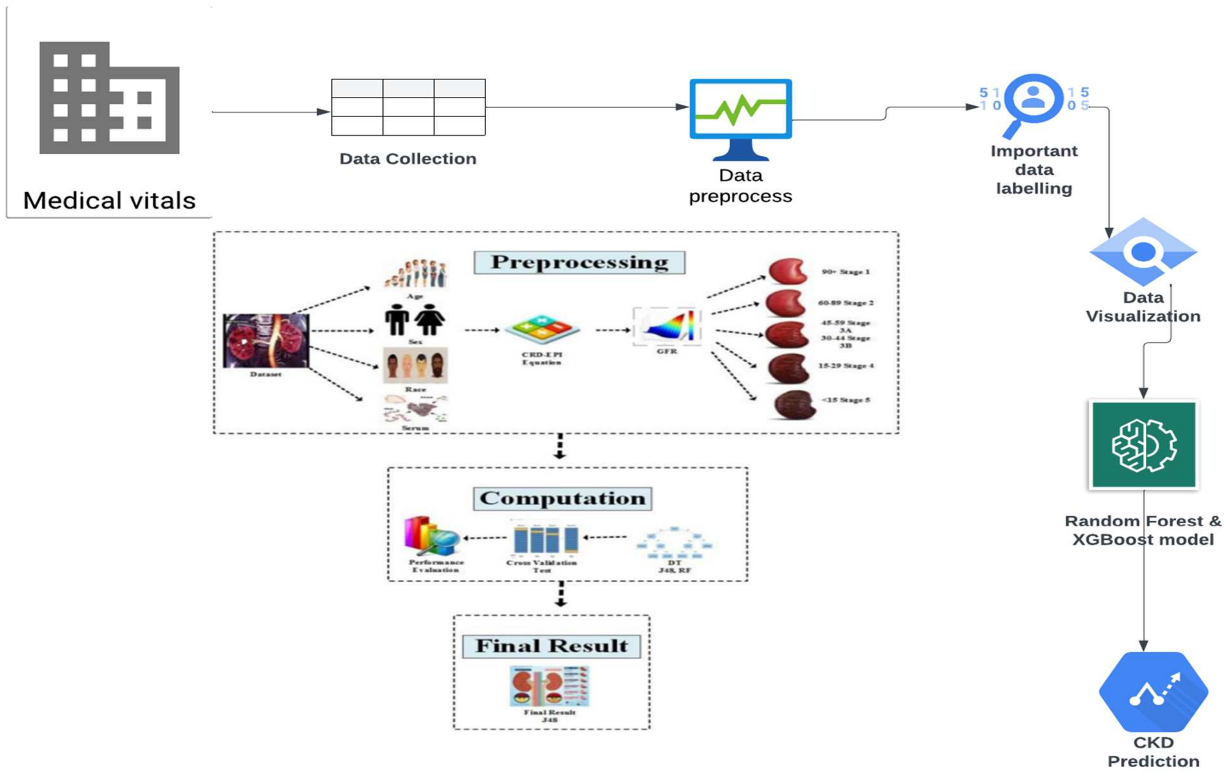


Fig 1: Chronic Kidney Disease Identification

Additionally, the paper reviews related works in the field of kidney cancer prediction, highlighting various machine learning approaches such as Neighborhood Component Analysis (NCA), Deep Neural Network (DNN), Hybrid Neural Network (HNN), and Recurrent Neural Network (RNN).

In summary, the research proposes an innovative approach using deep learning techniques for the prediction and diagnosis of chronic kidney disease. It demonstrates the potential of machine learning algorithms in improving early detection and management of kidney cancer, leveraging advancements in medical technology and data analytics.

Mithila rani et.al presented The research paper explores the prevalence and detection of Chronic Kidney Disease (CKD) using data mining techniques, specifically focusing on the Boruta algorithm.The Boruta algorithm, applied in this study, aids in identifying significant factors associated with CKD prediction. It works by creating shadow attributes and training a random forest classifier to assess attribute importance. Out of 24 attributes, only 7 were confirmed to be important in predicting CKD. The analysis shows that reducing the number of features can lead to a slightly lower accuracy of 99.19%, compared to 100% accuracy with all features, but significantly reduces processing time and memory load.

The research discusses the importance of factors such as hypertension, blood pressure, and specific tests like urine albumin and serum creatinine in detecting CKD. It emphasizes that a combination of tests is necessary for accurate diagnosis, especially in senior patients who are at higher risk. Furthermore, it suggests potential correlations between age and CKD-related factors.Numerical readings include statistics on attribute importance, with variables like sodium, age, packed cell volume, and hemoglobin deemed significant. Sensitivity and specificity of the model are reported as 1 and 0.98 respectively, with an out-of-bag estimate error rate of 1.08%. Confusion matrices illustrate the model's performance, with 47 correctly classified as "notCKD" and 75 as "CKD" in the reduced feature model.

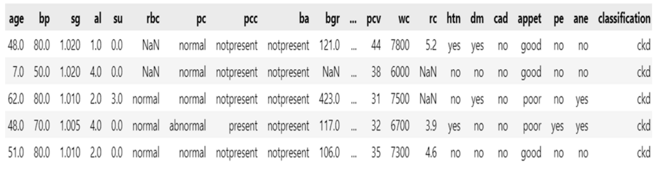
In conclusion, the study underscores the economic burden of CKD and the importance of affordable diagnostic methods. Boruta Analysis emerges as a valuable tool for medical diagnosis, offering cost-effective and faster solutions. The paper suggests avenues for future research, including the application of data mining algorithms in other chronic diseases for early detection and improved patient outcomes.

The Random Forest (RF) model exhibited the most superior predictive performance, boasting an average accuracy of 99.50%, sensitivity of 98.75%, specificity of 100%, precision of 100%, F1 score of 99.35%, and an AUC of 99.38%.Significant features highlighted by SHAP analysis included hemoglobin (hemo), specific gravity (sg), serum creatinine (sc), albumin (al), packed cell volume (pcv), red blood cell count (rbcc), hypertension (htn), blood glucose random (bgr), diabetes mellitus (dm), age, sodium (sod), blood urea (bu), and blood pressure (bp).

To streamline the dataset, a reduced set with 13 selected attributes was generated, leading to the formation of six distinct datasets based on various pathological tests. RF achieved the highest classification accuracy across different datasets, scoring 99.00% with the full dataset (DB-I), 97.75% with blood and other pathological tests (DB-II), and 97.00% with urine test attributes and others (DB-III).- GB and XGB classifiers also showed good performance, with RF generally outperforming LR and SVM.

III. METHODOLOGY

The research work focuses on identifying chronic kidney disease through an advanced methodology blending machine learning models. The methodology commences with comprehensive data collection, cleaning, and augmentation, followed by the division of the dataset into training, validation, and test sets. Feature extraction is performed using XGBoost, Random Forest and other models tailored for kidney disease identification are developed. The model is trained with advanced machine learning techniques, and advanced machine learning models like XGBoost , Forest tree classifier are explored. Ensemble methods and IoT platform experiments contribute to a holistic evaluation, with metrics such as accuracy, precision, recall, and F1 score. Results are analyzed, guiding fine-tuning and optimization for the development of an efficient and accurate model. Comprehensive documentation and reporting encompassing preprocessing, model architectures, and findings conclude the project. This methodology aims to advance early chronic kidney disease detection, emphasizing the ability of machine learning in healthcare.



age	bp	sg	al	su	rbc	pc	pcc	ba	bgr	...	pcv	wc	rc	htn	dm	cad	appet	pe	ane	classification
48.0	80.0	1.020	1.0	0.0	NaN	normal	notpresent	notpresent	121.0	...	44	7800	5.2	yes	yes	no	good	no	no	ckd
7.0	50.0	1.020	4.0	0.0	NaN	normal	notpresent	notpresent	NaN	...	38	6000	NaN	no	no	no	good	no	no	ckd
62.0	80.0	1.010	2.0	3.0	normal	normal	notpresent	notpresent	423.0	...	31	7500	NaN	no	yes	no	poor	no	yes	ckd
48.0	70.0	1.005	4.0	0.0	normal	abnormal	present	notpresent	117.0	...	32	6700	3.9	yes	no	no	poor	yes	yes	ckd
51.0	80.0	1.010	2.0	0.0	normal	normal	notpresent	notpresent	106.0	...	35	7300	4.6	no	no	no	good	no	no	ckd

Table 1: Chronic Kidney Disease Dataset

The dataset which is used for this research work consists of 24 features and 1 label. All features are basically human body vital datas which are directly and indirectly related for the cause of Chronic Kidney Disease. Label is categorized into two classes ckd and notCkd. The code begins by importing necessary libraries for data visualization, including pandas, numpy, matplotlib, seaborn, and plotly. It also suppresses warning messages and sets a specific plotting style. Then, the code mounts Google Drive to access data. After loading the dataset into a DataFrame (assumed as `df`), it converts specific columns to numerical type using `pd.to_numeric` with the `errors='coerce'` parameter to handle any conversion errors gracefully. Categorical and numerical columns are then separated into two lists using list comprehensions. The code iterates over categorical columns to print unique values for each column. This step is useful for understanding the nature and diversity of categorical data in the dataset. Overall, the code sets up the environment, prepares the dataset for analysis, and provides insights into the categorical data's uniqueness, which is crucial for subsequent analysis and visualization tasks.

$$f(x) \approx f(a) + f'(a)(x-a) + \frac{1}{2}f''(a)(x-a)^2$$

$$\mathcal{L}^{(t)} \simeq \sum_{i=1}^{n}[l(y_i, \hat{y}^{(t-1)}) + g_i f_t(\mathbf{x}_i) + \frac{1}{2}h_i f_t^2(\mathbf{x}_i)] + \Omega(f_t)$$

The top equation shows the Taylor Series expansion of the function f(x) around a base point a. This is a way of approximating a function with a polynomial centered around a specific input value.

The bottom equation shows part of the XGBoost objective function. In gradient boosting, the goal is to trim down a loss function over a set of training data. The loss function measures how different the predictions of a model are from the actual values. The XGBoost objective function combines a loss function, represented by l(yi,y ^(t−1)) (where i indexes the training sample, yi is the true value for that sample, and y^(t−1) is the prediction from the model at the previous iteration), with a regularization term, represented by Ω(fₜ). The regularization term penalizes the complexity of the model to prevent overfitting.

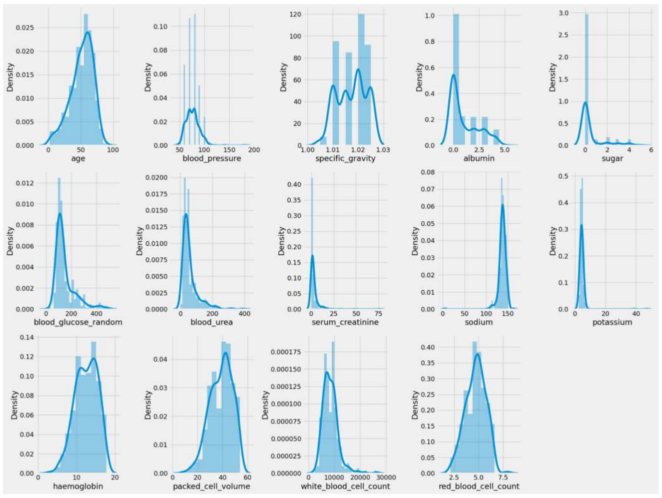


Fig 2: Distribution of all Numerical Features

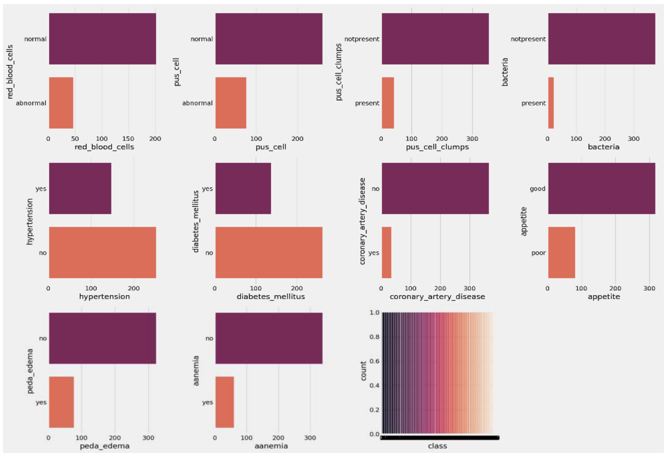


Fig 3: Distribution of all categorical Features

The code first creates subplots for visualizing the distribution of numerical and categorical columns separately. For numerical columns, it iterates over each column, plotting a distribution plot using Seaborn's `sns.distplot`. Similarly, for categorical columns, it plots a count plot using `sns.countplot`. The plots are arranged in a grid structure to fit the figure appropriately. Afterward, a heatmap of the correlation matrix is generated using Seaborn's `sns.heatmap`, annotating the correlations and displaying them color-coded. The heatmap is saved as a PNG file named 'corr1.png'.

Next, the code checks for missing values in the DataFrame using `df.isna().sum().sort_values(ascending=False)` to identify the columns with null values. It then defines two functions for

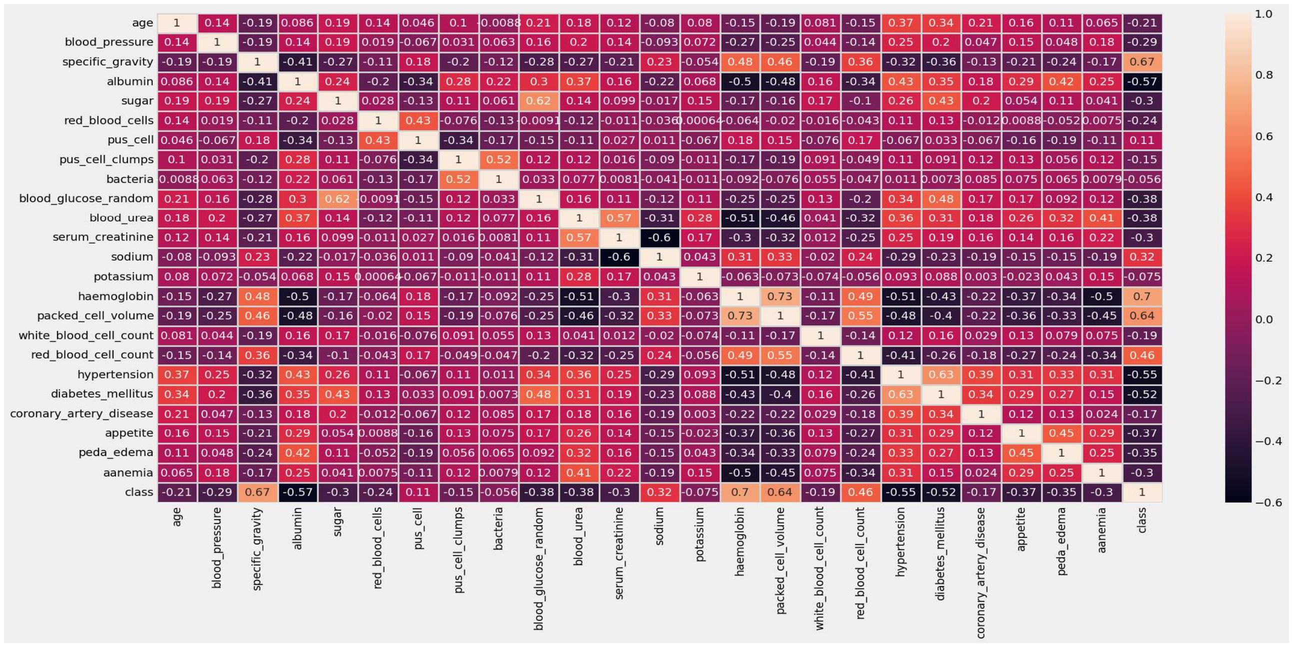


Fig 4: Heatmap of all Features of the Dataset

imputing missing values: `random_value_imputation` for numerical columns and 'red_blood_cells' and 'pus_cell' in categorical columns, and `impute_mode` for the rest of the categorical columns. Random sampling is used to fill null values in numerical columns, while mode imputation is used for categorical columns. The missing values are replaced accordingly.

Label encoding is applied to categorical columns using Scikit-learn's `LabelEncoder`, converting categorical values into numerical labels. Finally, correlations with the target variable 'class' are computed and displayed for both numerical and categorical features. This comprehensive approach ensures the dataset is prepared for further analysis and modeling by handling missing values and encoding categorical features appropriately.

The code begins by importing necessary libraries for scaling features, splitting data, and implementing machine learning algorithms. It uses `MinMaxScaler` from Scikit-learn to scale up the features to a specified range. The scaled features are stored in `new_features` after transforming the original feature matrix `X`. The data is then split into training and test sets using `train_test_split` from Scikit-learn, with 75% of the data used for training and 25% for testing, maintaining consistency with a random state of 67 for reproducibility.

Five classification models are trained and evaluated: K-Nearest Neighbors (KNN), Random Forest Classifier, Decision Tree Classifier, Gradient Boost Classifier and XGBoost. For each model, the training and test accuracies are printed along with a classification report, which provides metrics like precision, recall, and F1-score for each class. Additionally, confusion matrices are plotted using a custom function `plot_confusion_matrix`. After fitting each model to the training data, the accuracies and classification reports for the test data are printed out. These metrics provide insights into how precise each model performs on unseen data. The final step involves displaying the confusion matrices for each model, allowing for a visual examination of the model's performance in classifying different classes.

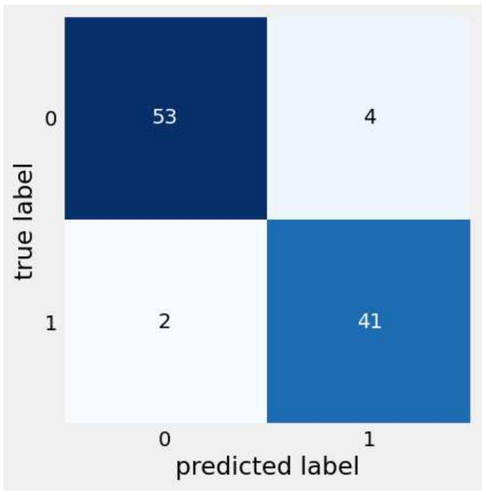


Fig 5: Confusion Matrix of KNN model

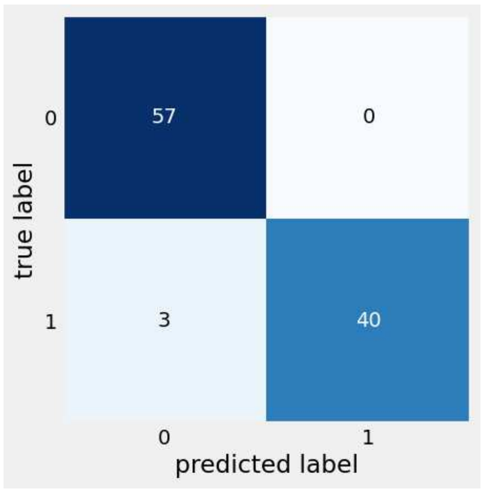


Fig 6: Confusion Matrix of Random Forest model

The Random Forest, Gradient Boost, and XGBoost classifiers achieved perfect training accuracy but showed slightly lower test accuracies of 0.97, 0.97, and 0.95, respectively, indicating a minor drop in performance on unseen data. Precision, recall, and F1-score metrics were high for both classes, suggesting few false positives and robust performance in identifying instances of both classes. Despite the slight decrease in test accuracy, the Random Forest Classifier displayed strong overall performance in classifying the dataset, with high precision, recall, and F1-scores across all classes.

IV. RESULTS AND DISCUSSION

Chronic Kidney disease Identification using various advanced machine learning models have revolutionized result and implementation in the filed of medical diagnosis. This research work's performance was measured by three metrics: precision, recall and f1-score. Early Detection of CKD can be very helpful in order to enable timely intervention and personalized treatment plans.

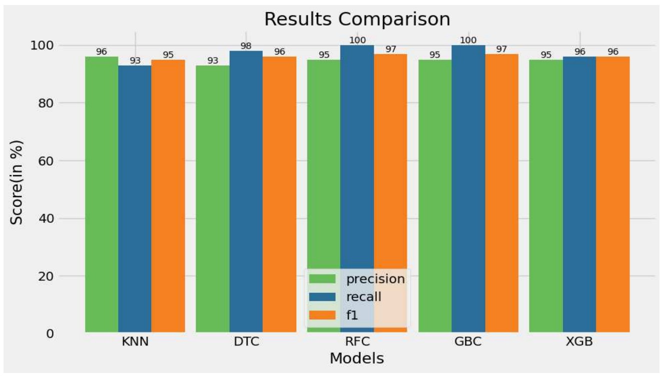


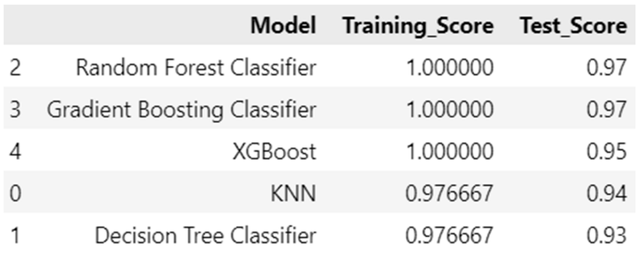
Fig. 7: Performance Measures of Various Predictive Models

The K-Nearest Neighbors (KNN) classifier attained a training accuracy of 97.67% and a test accuracy of 94.00%. Precision, recall, and F1-score for both negative (class 0) and positive (class 1) instances were reported. For class 0, precision was 96%, recall was 93%, and F1-score was 95%. For class 1, precision was 91%, recall was 95%, and F1-score was 93%. These metrics collectively indicate the classifier's balanced performance in distinguishing between the two classes. With an overall accuracy of 94%, the KNN model demonstrated effectiveness in accurate predictions on the test dataset.

The Decision Tree Classifier (DTC) achieved perfect training accuracy of 100% and 95% test accuracy. Precision, recall, and F1-score were reported for both negative (class 0) and positive (class 1) instances. For class 0, precision was 93%, recall was 98%, and F1-score was 96%. For class 1, precision was 97%, recall was 91%, and F1-score was 94%. These metrics collectively demonstrate the classifier's strong performance in differentiating between the two classes. With an overall accuracy of 95%, the Decision Tree model exhibited robust predictive power on the test dataset.

The Random Forest Classifier achieved 100% training accuracy and a high 97% test accuracy. Precision, recall, and F1-score were reported for both negative (class 0) and positive (class 1) instances. For class 0, precision was 95%, recall was 100%, and F1-score was 97%. For class 1, precision was 100%, recall was 93%, and F1-score was 96%. These metrics collectively demonstrate the classifier's robust performance in distinguishing between the two classes. With an overall accuracy of 97%, the Random Forest model showcases strong predictive capability on the test dataset.

XGBoost classifier is performing exceptionally well! The test accuracy of 0.95 means it's highly effective at classifying new data. The classification report confirms this, showing excellent precision of 95%, recall 96%, & 93% and f1-scores 96% & 94% across both classes. However, the perfect training accuracy raises a potential concern of overfitting. This means the model might be overly-tuned to the training data, so keep an eye on its performance with a separate validation dataset to ensure it generalizes well to unseen examples.



	Model	Training_Score	Test_Score
2	Random Forest Classifier	1.000000	0.97
3	Gradient Boosting Classifier	1.000000	0.97
4	XGBoost	1.000000	0.95
0	KNN	0.976667	0.94
1	Decision Tree Classifier	0.976667	0.93

Table 2: Training & Test Accuracies of Various Models

The table summarizes the performance of different machine learning models. XGBoost, Random Forest Classifier and Gradient Boosting Classifier achieved perfect training scores of 100% and test scores of 97%. XGBoost demonstrated a slightly lower test score of 95% despite a perfect training score. KNN achieved a training accuracy of about 97.67% with a corresponding test accuracy of 94%. The Decision Tree Classifier showed a test score of 93% with a similar training accuracy to KNN. Overall, Random Forest Classifier and Gradient Boosting Classifier performed best, closely followed by XGBoost, KNN, and the Decision Tree Classifier.

Evaluation indicators after developing the confusion matrix:

$$Accuracy\ Score = \frac{TP+TN}{TP+TN+FP+FN}$$

$$Precision\ Score = \frac{TP}{TP+FP}$$

$$Recall\ Score = \frac{TP}{TP+FN}$$

$$F1\ Score = \frac{2*(Precision*Recall)}{Precision+Recall}$$

Model	Precision	Recall	F1-score	Accuracy
KNN	0.96	0.93	0.95	0.94
Decision Tree Classifier	0.93	0.98	0.96	0.93
Random Forest Classifier	0.95	1.00	0.97	0.97
Gradient Boost Classifier	0.95	1.00	0.97	0.97
XGBoost	0.95	0.96	0.96	0.95

Table 3: Performance Measures of all Models

A. Advanced ML Models: Explore and implement advanced machine learning models to enhance the accuracy and reliability of CKD predictions, ensuring robust performance in diverse healthcare scenarios.

B. Feature Processing and conversion: Develop data preprocessing environment to process features dimensions and converting categorical data into numerical form improving the efficiency and accuracy of the CKD prediction system.

C. Multi model usage: Implementing multiple machine learning models that not only identifies chronic kidney disease (CKD) but also provides insights also considering individual health data, lifestyle factors, and genetic predispositions.

V. CONCLUSION AND FUTURE WORK

In conclusion, this research work has sought to address the critical issue of Chronic Kidney Disease (CKD) identification through the implementation of advanced Machine Learning (ML) models, with a specific focus on the novel XGBoost, Random forest. The project was driven by the recognition of the life-threatening nature of CKD, the challenges in early detection, and the potential of cutting-edge technology to enhance diagnostic accuracy. The highest accuracy is of the Random forest & gradient boost model which is 97% both. The project's primary objectives were to develop a robust ML-based system for early CKD identification, leverage the capabilities of the XGBoost and Random Forest, and contribute to the evolving landscape of healthcare through the integration of intelligent solutions and webapp functionality.

A. Personalized Medicine - Developing models that consider individual patient characteristics, genetic factors, lifestyle, and environmental influences for personalized CKD risk assessment and treatment plans.

B. Remote Patient Monitoring - Designing applications and wearable devices that enable remote monitoring of patients with CKD, allowing healthcare professionals to track key indicators and intervene as needed. Integrating patient-specific data to tailor interventions and medication regimens for better outcomes.

C. Disease Progression Prediction - Build an AI model capable of predicting the progression of diseases detected through patient data. This information could guide treatment plans and help healthcare providers anticipate and address potential complications.

REFERENCES

[1] Guozhen chen et al. Prediction of Chronic Kidney Disease Using Adaptive Hybridized Deep Convolutional Neural Network on the Internet of Medical Things Platform , IEEE Access 1109, may 18, 2020

[2] PEDRO A. MORENO-SÁNCHEZ “Data-Driven Early Diagnosis of Chronic Kidney Disease: Development and Evaluation of an Explainable AI Model ” IEEE Access 3rd April, 2023

[3] Sagar Dhanraj Pande et al. “Multi-Class Kidney Abnormalities Detecting Novel System Through Computed Tomography” IEEE Access vol 45 no 2nd June, 2020

[4] Ping Liang et al. “Deep Learning Identifies Intelligible Predictors of Poor Prognosis in Chronic Kidney Disease”, EMB vol 27 , 7th July, 2023

[5] MD. RASHED-AL-MAHFUZ et al. “Clinically Applicable Machine Learning Approaches to Identify Attributes of Chronic Kidney Disease (CKD) for Use in Low-Cost Diagnostic Screening” IEEE Healthcare vol 11 no 25 15th April, 2021.

[6] Maithaili desai “Early Detection and Prevention of Chronic Kidney Disease” vol 21 no 70 3rd May, 2021

[7] Asif Salekin et al . “Detection of Chronic Kidney Disease and Selecting Important Predictive Attributes”

2016 IEEE International Conference on Healthcare Informatics access vol 22 no 17 5th March, 2021

[8] Dr. Razib Hayat Khan et al. “A Comparative Analysis of Machine Learning Approaches for Chronic Kidney Disease Detection” 023 8th International Conference on Electrical, Electronics and Information Engineering (ICEEIE) vol 23 no 31 May 2023

[9] Dina saif et al. “Deep-kidney: an effective deep learning framework for chronic kidney disease prediction” Vol 21 no 13, March 2019

[10] Rahul Sawhney et al. “A comparative assessment of artificial intelligence models used for early prediction and evaluation of chronic kidney disease.” Elsevier - Decision Analytics Journal 6 vol 21 no 18 (2023) 100169.

[11] Md. Ariful Islam et al. "Chronic kidney disease prediction based on machine learning algorithms" J Pathol Inform. 2023; vol 14: 100189.

[12] Dibaba Adeba Debal et al. "Chronic kidney disease prediction using machine learning techniques. Vol 21 Article number: 109 may (2022) "

[13] Hira Khalid et al. "Machine Learning Hybrid Model for the Prediction of Chronic Kidney Disease Published vol 25 no 4, 14 Mar 2023 "

[14] Imesh Udara et al. "Chronic Kidney Disease Prediction Using Machine Learning Methods vol 28 no 21, 28-30 July 2020"

[15] Deema Mohammed Alsekait et al. "Toward Comprehensive Chronic Kidney Disease Prediction Based on Ensemble Deep Learning Models, vol 20 no 21, 20 March 2023".

[16] Qiong Bai et al. "Machine learning to predict end stage kidney disease in chronic kidney disease, vol 12, Article number: 8377 (2022)"

[17] Nikhila et al. "Chronic Kidney Disease Prediction using Machine Learning Ensemble Algorithm vol 15 no 19, 19-20 February 2021"

[18] Reshma S et al. "Chronic Kidney Disease Prediction using Machine Learning, Vol. 9 Issue 07, July-2020"

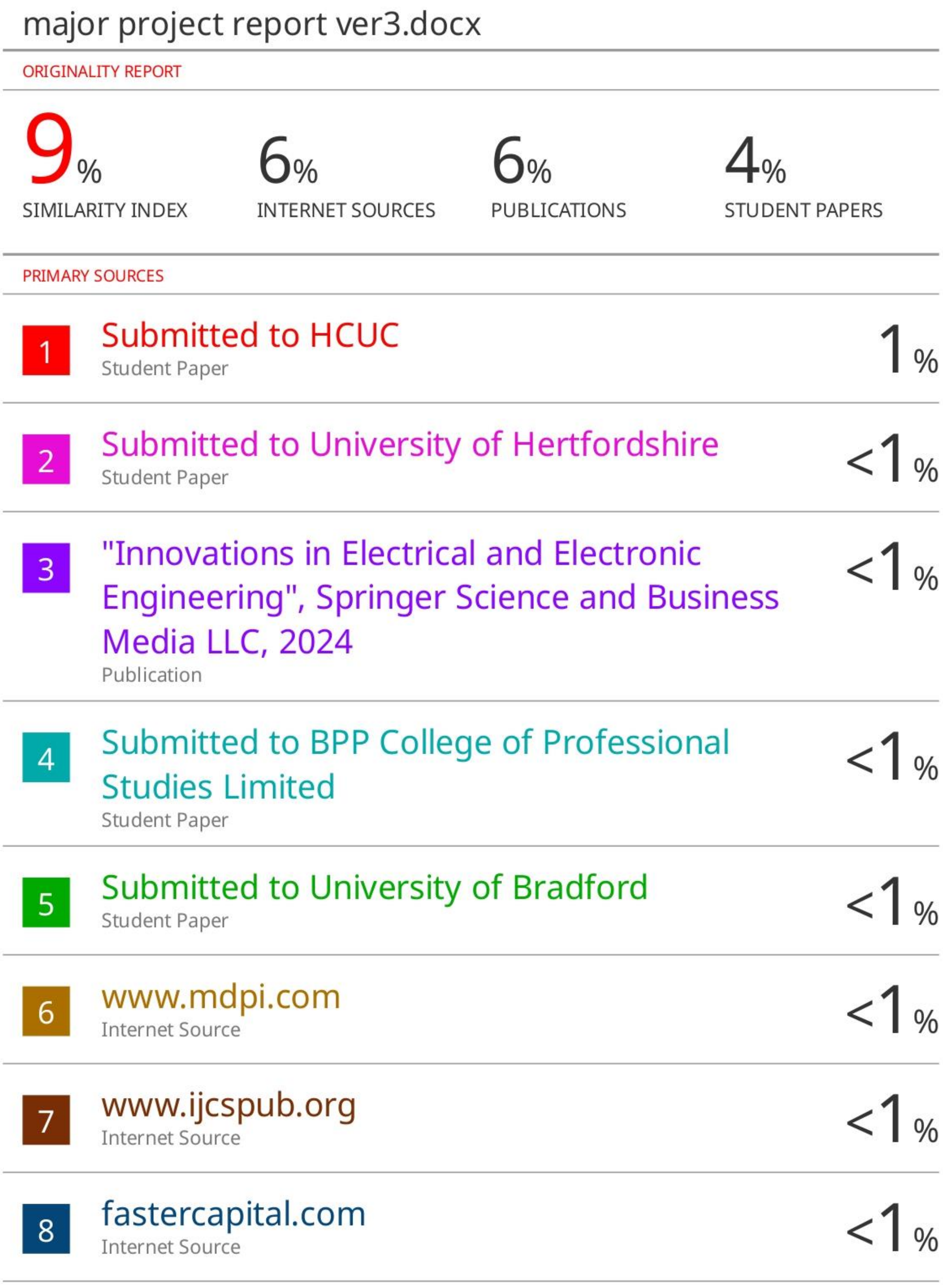
[19] Marwa Almasoud et al. "Detection of Chronic Kidney Disease Using Machine

Learning Algorithms with Least Number of Predictors, vol 8 issue 21, march 2013"

[20] Chamandeep Kaur et al. "Chronic Kidney Disease Prediction Using Machine Learning, ournal of Advances in Information Technology, Vol. 14, No. 2, 2023"

[21] Md. Mehedi Hassan et al. "A Comparative Study, Prediction and Development of Chronic Kidney Disease Using Machine Learning on Patients Clinical Records, vol 12 no 18, 22 February 2023.

APPENDIX
D PLAGIARISM REPORT



major project report ver3.docx

ORIGINALITY REPORT

9%	6%	6%	4%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to HCUC Student Paper	1%
2	Submitted to University of Hertfordshire Student Paper	<1%
3	"Innovations in Electrical and Electronic Engineering", Springer Science and Business Media LLC, 2024 Publication	<1%
4	Submitted to BPP College of Professional Studies Limited Student Paper	<1%
5	Submitted to University of Bradford Student Paper	<1%
6	www.mdpi.com Internet Source	<1%
7	www.ijcspub.org Internet Source	<1%
8	fastercapital.com Internet Source	<1%

9	egusphere.net Internet Source	<1 %
10	Maithili Desai. "Early Detection and Prevention of Chronic Kidney Disease", 2019 5th International Conference On Computing, Communication, Control And Automation (ICCUBEA), 2019 Publication	<1 %
11	Sagar Dhanraj Pande, Raghav Agarwal. "Multi-Class Kidney Abnormalities Detecting Novel System Through Computed Tomography", IEEE Access, 2024 Publication	<1 %
12	Submitted to Brunel University Student Paper	<1 %
13	Abhinav Srivastava, Shloka Samanta, Sushruta Mishra, Ahmed Alkhayyat, Deepak Gupta, Vandana Sharma. "Medi-Assist: A Decision Tree based Chronic Diseases Detection Model", 2023 4th International Conference on Intelligent Engineering and Management (ICIEM), 2023 Publication	<1 %
14	Anamika Paul Rupa, Aryya Gangopadhyay. "Multi-modal Deep Learning Based Fusion Approach to Detect Illicit Retail Networks from Social Media", 2020 International	<1 %

	Conference on Computational Science and Computational Intelligence (CSCI), 2020 Publication	
15	Md. Rashed-Al-Mahfuz, Abedul Haque, AKM Azad, Salem A. Alyami, Julian M.W. Quinn, Mohammad Ali Moni. "Clinically applicable machine learning approaches to identify attributes of Chronic Kidney Disease (CKD) for use in low-cost diagnostic screening", IEEE Journal of Translational Engineering in Health and Medicine, 2021 Publication	<1%
16	Submitted to The Robert Gordon University Student Paper	<1%
17	library.iugaza.edu.ps Internet Source	<1%
18	Submitted to Aston University Student Paper	<1%
19	penerbit.uthm.edu.my Internet Source	<1%
20	iugspace.iugaza.edu.ps Internet Source	<1%
21	B Anjali, R Reshma, V Geetha Lekshmy. "Detection of Counterfeit News Using Machine Learning", 2019 2nd International Conference on Intelligent Computing,	<1%

	Instrumentation and Control Technologies (ICICICT), 2019 Publication	
22	medium.com Internet Source	<1%
23	www.oajaiml.com Internet Source	<1%
24	Ubaida Fatima, Sadia Kiran, Muhammad Fouzan Akhter, Muhammad Kumail, Jaweria Sohail. "Unveiling the Optimal Approach for Credit Card Fraud Detection: A Thorough Analysis of Deep Learning and Machine Learning Methods", Research Square Platform LLC, 2024 Publication	<1%
25	hrcak.srce.hr Internet Source	<1%
26	unsworks.unsw.edu.au Internet Source	<1%
27	arxiv.org Internet Source	<1%
28	www.researchgate.net Internet Source	<1%
29	Submitted to University of Sunderland Student Paper	<1%
	Submitted to Heriot-Watt University	

30	Student Paper	<1%
31	www.coursehero.com Internet Source	<1%
32	Submitted to University of Greenwich Student Paper	<1%
33	easpublisher.com Internet Source	<1%
34	www.kdnuggets.com Internet Source	<1%
35	www.linuxpromagazine.com Internet Source	<1%
36	Submitted to Bournemouth University Student Paper	<1%
37	spiegato.com Internet Source	<1%
38	www.ijraset.com Internet Source	<1%

Exclude quotes	On	Exclude matches	< 10 words
Exclude bibliography	On		