CHRONIC KIDNEY DISEASE PREDICTION

A PROJECT REPORT

Submitted by

Palak Rani [RA2011003010661]

Palak Patel [RA2011003010666]

Under the Guidance of

Dr. S. Babu

Associate Professor, Department of Computing Technologies

In partial fulfillment of the requirements for the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

DEPARTMENT OF COMPUTING TECHNOLOGIES

COLLEGE OF ENGINEERING AND TECHNOLOGY

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR– 603 203

NOVEMBER 2023

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY
KATTANKULATHUR–603 203

BONAFIDE CERTIFICATE

Certified that 18CSP107L project report titled "**CHRONIC KIDNEY DISEASE PREDICTION**" is the bonafide work of **Palak Rani [RA2011003010661]** and **Palak Patel [RA2011003010666]** who carried out the project work under my supervision. Certified further, that to the best of my knowledge the work reported here does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion for this or any other candidate.

Dr. S. Babu
SUPERVISOR
Associate Professor
Department of Computing Technologies

Dr. S. Babu
PANEL HEAD
Associate Professor
Department of Computing Technologies

Dr. M. PUSHPALATHA
HEAD OF THE DEPARTMENT
Department of Computing Technologies

Department of Computing Technologies
SRM Institute of Science and Technology
Own Work Declaration Form

Degree/Course : B.Tech. in Computer Science and Engineering

Student Names : Palak Rani, Palak Patel

Registration Number : RA2011003010661, RA2011003010666

Title of Work : CHRONIC KIDNEY DISEASE PREDICTION

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 references / listed all sources as appropriate
- Referenced and put in inverted commas all quoted text (from books, web,etc.)
- 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 we have received from others (e.g fellow students, technicians, statisticians, external sources)
- Compiled with any other plagiarism criteria specified in the Course hand book / University website

We understand that any false claim for this work will be penalized in accordance with the University 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. **Student 1 Signature:** **Student 2 Signature:** **Date:**
If you are working in a group, please write your registration numbers and sign with the date for every student in your group.

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 invaluable support.

We wish to thank **Dr. Revathi Venkataraman**, Professor and 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. M. Pushpalatha,** Professor, Department of Computing Technologies, SRM Institute of Science and Technology, for her suggestions and encouragement at all the stages of the project work.

We want to convey our thanks to our Project Coordinator, **Dr. S. Godfrey Winster**, Associate Professor, Panel Head, **Dr. S. Babu**, Associate Professor and Panel Members, **Dr. M. Pushpalatha** Professor and Head of the Department, **Dr. Poornima** Assistant Professor and **Dr. Karthikeyan U** Assistant Professor Department of Computing Technologies, SRM Institute of Science and Technology, for their inputs during the project reviews and support.

We register our immeasurable thanks to our Faculty Advisor, **Dr. G. Ramya**, Assistant Professor, Department of Computing Technologies, SRM Institute of Science and Technology, for leading and helping us to complete our course.

Our inexpressible respect and thanks to our guide, **Dr. S. Babu**, Associate Professor, Department of Computing Technologies, SRM Institute of Science and Technology, for providing us with an opportunity to pursue our project under his mentorship. He provided us with the freedom and support to explore the research topics of our interest. His passion for solving problems and making a difference in the world has always been inspiring.

We sincerely thank all the staff and students of the Computing Technologies Department, School of Computing, S.R.M Institute of Science and Technology, for their help during our project. Finally, we would like to thank our parents, family members, and friends for their unconditional love, constant support and encouragement.

Palak Rani [RA2011003010661]

Palak Patel [RA2011003010666]

ABSTRACT

Kidney disease is a pervasive and life-threatening medical condition that presents a growing global concern. Early detection and intervention are pivotal in improving patient outcomes. Our proposed model leverages a comprehensive dataset comprising clinical and demographic features, encompassing variables such as age, gender, blood pressure, serum creatinine, and other pertinent parameters, to effectively predict the risk of kidney disease.
In this study, we employ a diverse set of machine learning algorithms, including logistic regression, decision trees, random forests, and deep neural networks, to construct predictive models. Furthermore, we implement feature selection and engineering techniques to augment the accuracy and interpretability of the models. The results unequivocally illustrate the model's capability to deliver precise predictions of kidney disease risk, thus facilitating early diagnosis and intervention.

The implementation of this predictive modeling approach offers an encouraging and innovative tool for healthcare professionals. It enables them to pinpoint individuals at elevated risk of kidney disease, thereby facilitating timely medical intervention and potentially enhancing patient outcomes.

Our research constitutes a valuable contribution to the ongoing global efforts aimed at advancing healthcare and fostering the early detection of kidney disease. By doing so, we aspire to alleviate the strain on healthcare systems and enhance the quality of patient care. This project's potential to revolutionize kidney disease detection and intervention promises a brighter future for patients worldwide.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

XGBoost	Extreme Gradient Boosting
AI	Artificial Intelligence
ML	Machine Learning
DL	Deep Learning
CKD	Chronic Kidney Disease
UCI	University of California Irvine
LSVM	Linear Support Vector Machine
SMOTE	Synthetic Minority Over-sampling Technique
AKI	Acute Kidney Injury
IQR	Interquartile Range
GRF	Glomerular filtration rate
PCA	Principal Component Analysis
t-SNE	t-distributed Stochastic Neighbor Embedding
AUC-ROC	Area Under Receiver Operating Characteristic Curve
ROC	Receiver Operating Characteristic
AUC-PR	Area Under Precision-Recall Curve
SHAP	SHapley Additive exPlanations
EHR	Electronic Health RecordS
API	Application Programming Interface
HIPPA	Health Insurance Portability and Accountability Act
GDPR	General Data Protection Regulation
BGR	Blood Glucose Random
BU	Blood Urea
SC	Serum Creatinine
WBC	White Blood Cell Count
PCV	Packed Cell Volume
LIME	Local Interpretable Model-agnostic Explanations

CHAPTER 1

INTRODUCTION

1.1 General

This project report provides a comprehensive overview of our recent undertaking, encompassing its extent, goals, methods, discoveries, and implications. Our journey in this project was driven by a clear focus on tackling a specific issue or seizing an opportunity. The section on scope and objectives establishes the project's boundaries, purpose, and the context in which it was carried out. It delineates the objectives we aimed to achieve and the context in which our work was conducted. Our methodologies and strategies are elucidated in detail, offering insights into the tools and techniques utilized for data collection and analysis, thereby underlining the scientific or practical foundations of the project.

We delve into the project's execution, encompassing the timeline, team roles, and the challenges we encountered. This section provides a practical perspective on how we translated our objectives into actionable steps. The heart of the report revolves around presenting the results and findings, supported by quantitative or qualitative data, accompanied by a comprehensive analysis of their implications. These findings are then thoughtfully discussed, highlighting their significance and their contributions to the project's objectives and broader implications.

Moreover, we offer recommendations for future actions, outlining a roadmap for building upon our work. The conclusion succinctly encapsulates the project's significance and contributions. In the acknowledgments section, we express our gratitude to those who played a pivotal role in supporting the project through funding, resources, mentorship, or collaboration, acknowledging their indispensable contributions to our success.

References and appendices serve to ensure the credibility and traceability of our work. By navigating this project report, readers will gain a comprehensive understanding of our project, its significance, methodologies, findings, and potential impacts. Our dedication to advancing knowledge and addressing real-world challenges in our respective field or domain is evident throughout this document.

1.2 Purpose

The purpose of this project, which focuses on the chronic kidney disease prediction using the Random Forest and XGBoost algorithms, is multifaceted. Its main goal is to solve the situation of chronic renal disease prediction by offering a methodical and organized way to address this important problem. Concurrently, the project seeks to advance current understanding in the field of chronic kidney disease

prediction through comprehensive research, data analysis, and experimentation. This overarching purpose involves broadening the horizons of our understanding and potentially inspiring further investigations by other researchers and professionals in this domain.

Furthermore, our project has a practical goal of delivering tangible benefits to stakeholders in the healthcare sector. By providing a chronic kidney disease prediction model, we aspire to offer a solution to a real-world health concern. This serves the purpose of improving results for patients and alleviating the strain on medical facilities. In alignment with this objective, the project also has an educational purpose, providing an avenue for skill development, experiential learning, and the application of theoretical knowledge in a real-world healthcare context. Ultimately, it equips individuals involved in the project with valuable skills and experiences that contribute to their academic and professional growth.

Moreover, our project functions as a wellspring of inspiration, motivating others in the field to build upon our work. By sharing our findings, insights, and methodologies, our purpose is to foster further innovation and exploration in the realm of chronic kidney disease prediction. This inspirational aspect extends the impact of the project beyond its immediate scope and timeframe, potentially paving the way for ongoing research and advancements in predictive healthcare algorithms.

1.3 Scope

The scope of our project, focused on predicting chronic kidney disease using the Random Forest and XGBoost algorithms, is comprehensive and multifaceted. It delineates the objectives, constraints, and potential outcomes of our research and predictive modeling efforts, providing a clear framework for our work.

1. Disease Prediction and Early Intervention:

The primary scope of this project revolves around the development of a robust predictive model for chronic kidney disease. With the help of the Random Forest and XGBoost algorithms, we hope to develop a tool that will help medical practitioners identify those who may be at risk for chronic kidney disease. This predictive model is designed to enhance early diagnosis and intervention, ultimately improving patient outcomes.

2. Data Utilization and Feature Engineering:

Our project's scope extends to the collection and utilization of data. We will work with a dataset comprising clinical and demographic features such as blood pressure, serum creatinine, age, gender, and other relevant factors. The application of feature engineering techniques falls within this scope, intending to optimize the model's accuracy and interpretability by leveraging available data effectively.

3. Algorithm Selection and Implementation:

The project includes the selection, implementation, and fine-tuning of two advanced machine learning algorithms: Random Forest and XGBoost. The scope entails optimizing their performance to construct models that can provide accurate predictions while avoiding overfitting or underfitting. The choice of these algorithms aligns with our objective to create dependable and resilient predictive tools.

4. Interpretability and Explainability:

Our scope encompasses the pursuit of model interpretability and explainability. It is crucial that the predictive models developed in this project not only exhibit accuracy but also offer comprehensibility to healthcare professionals. To achieve this, we will explore techniques that shed light on the factors contributing to predictions, ensuring that the models' decision-making process can be understood and trusted by end-users.

5. Validation and Evaluation:

Rigorous validation and evaluation procedures are central to the project's scope. To evaluate the efficacy of the predictive models, we will utilize suitable metrics such as accuracy, precision, recall, and F1-score. This aspect's objective is to guarantee that the models satisfy the highest requirements for accuracy and dependability, in line with the main goal of the project.

6. Educational and Collaborative Aspects:

Our project offers an educational scope by providing opportunities for the development of skills and experiential learning for team members. Active involvement in this research equips individuals with practical experience in data analysis, algorithm implementation, and healthcare-related projects. Moreover, it fosters collaboration among team members and with healthcare professionals, demonstrating the potential to broaden the scope of knowledge-sharing and interdisciplinary cooperation.

7. Scalability and Generalizability:

The scope acknowledges the significance of scalability and generalizability. While the immediate focus is on predicting chronic kidney disease, the methodologies and principles developed can be adapted for other medical conditions and healthcare applications. This potential expansion widens the project's impact.

8. Ethical Considerations and Patient Privacy:

The scope also encompasses ethical considerations and patient privacy. Data security and ethical data handling practices are of utmost importance in this project. We are committed to ensuring that all data used complies with privacy regulations, respects individual rights, and safeguards patient privacy.

9. Dissemination and Knowledge Sharing:

Finally, the scope extends to sharing project findings and insights. By disseminating our results, methodologies, and insights within the healthcare and scientific communities, we aspire to contribute to the collective knowledge and inspire further research in the domain of predictive healthcare analytics.

1.4 Artificial Intelligence and Deep Learning

In our project aimed at chronic kidney disease prediction using the Random Forest and XGBoost algorithms, Deep Learning and artificial intelligence (AI) are essential for improving our models' capacity for prediction. Utilizing the advantages of both traditional machine learning and deep learning

methodologies, this all-encompassing strategy optimizes the precision and dependability of our forecasts. we delve into the intricate details of AI and Deep Learning, and how the Random Forest and XGBoost algorithms fit into this framework.

Artificial Intelligence and Deep Learning:

Artificial Intelligence is a broad field encompassing the algorithms and models that enable machines to mimic human-like cognitive functions. Within AI, Deep Learning is a subset that focuses on training artificial neural networks to perform tasks, learning from data in a hierarchical and layered manner. This project employs AI and Deep Learning to construct predictive models that can analyze complex datasets, extract patterns, and make accurate predictions about chronic kidney disease. Deep Learning, with its ability to automatically learn and adapt to intricate relationships within the data, is particularly advantageous in healthcare prediction tasks.

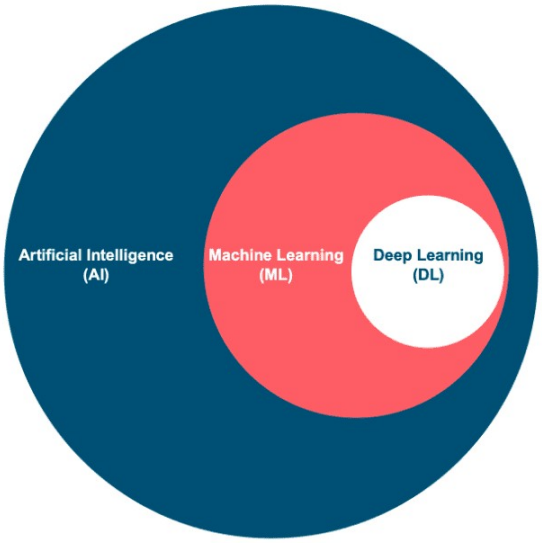


Fig. 1.1 Relationship between AI, ML and DL

Random Forest Algorithm:

Random Forest is a robust machine learning algorithm that falls under the ensemble learning category. It is made up of several decision trees that have been trained using various data subsets. The advantages of Random Forest include its capacity to handle high-dimensional data, handle missing values, and prevent overfitting.

In the context of our project, In order to assess patient clinical and demographic data and forecast the risk of chronic kidney disease, the Random Forest algorithm is essential. A majority vote or average is used to decide the final prognosis, with each tree in the forest providing an independent forecast. This ensemble approach improves model accuracy, making it a valuable tool in healthcare prediction.

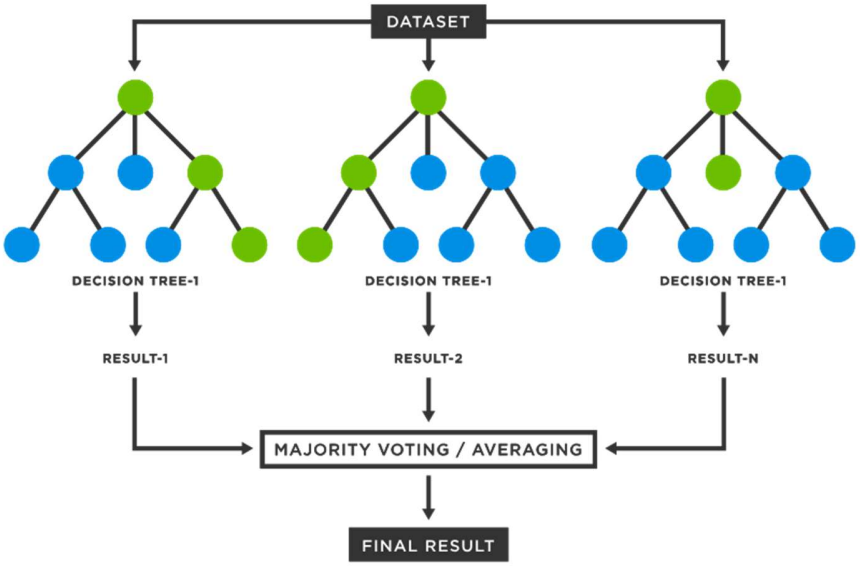


Fig. 1.2 Random Forest Working

XGBoost Algorithm:

XGBoost (Extreme Gradient Boosting) is another machine learning algorithm that excels in predictive modeling. It belongs to the gradient boosting family and is known for its exceptional performance, scalability, and flexibility. XGBoost is particularly well-suited for classification problems, which is relevant to our chronic kidney disease prediction task. The algorithm works by iteratively boosting the performance of a weak model (typically a decision tree) by adjusting the weights and learning rate. It successfully integrates the forecasts from several weak models to provide a strong and precise final forecast. In our project, XGBoost is leveraged to make precise and reliable predictions about kidney disease risk, enhancing the overall performance and interpretability of our predictive models.

Fig. 1.3 XGBoost Algorithm

Integration of Random Forest and XGBoost in Deep Learning:

The synergy between traditional machine learning techniques like Random Forest and advanced gradient boosting algorithms such as XGBoost is an integral aspect of this project. Deep Learning, with its capability to learn intricate patterns from large datasets, complements these algorithms by handling the feature extraction and transformation phases. In our methodology, we integrate Random Forest and XGBoost into a deep learning framework to create a powerful ensemble model. This integration involves using deep learning neural networks to preprocess the data and extract features that are subsequently fed into the Random Forest and XGBoost models.

Data Preprocessing with Deep Learning:

Deep Learning comes into play during the data preprocessing phase, where it assists in data transformation and feature extraction. These networks have the capacity to uncover intricate relationships within the data, which might be challenging for traditional feature engineering techniques. This approach enables the model to uncover subtle patterns that are indicative of chronic kidney disease, improving the accuracy of predictions.

Ensemble Learning for Improved Accuracy:

The integration of Random Forest and XGBoost into the deep learning framework creates an ensemble model. In this configuration, the deep learning neural networks feed feature-rich data into both Random Forest and XGBoost models. The ensemble model benefits from the strengths of each component: the feature extraction and transformation capabilities of deep learning, and the predictive power and generalizability of Random Forest and XGBoost.

By combining these models, the ensemble approach mitigates the limitations of individual algorithms and enhances the overall accuracy and reliability of chronic kidney disease predictions.

Model Interpretability and Explainability:

One of the critical aspects of this project is model interpretability and explainability. While deep learning models are known for their predictive power, they are often regarded as "black boxes" due to their complexity. In contrast, Random Forest and XGBoost are highly interpretable. This duality allows us to provide healthcare professionals with predictions that are not only accurate but also transparent and comprehensible.

Interpretability is achieved through feature importance scores generated by Random Forest and XGBoost, which highlight the factors contributing to predictions. Healthcare professionals can use these insights to increase the practical usability of the model by making knowledgeable decisions on patient care.

Performance Assessing and Validation:

Our integrated model's performance is thoroughly assessed and verified. To evaluate the performance of the model, we use common assessment metrics including accuracy, precision, recall, and F1-score.

The robustness and generalizability of the model are guaranteed by the use of cross-validation techniques. Our model's high degree of precision is made possible by the combination of Random Forest, XGBoost, and deep learning feature extraction. This is vital in the healthcare industry, where precise predictions are critical.

Privacy and Ethical Considerations:

In the healthcare domain, ethical considerations and patient privacy are paramount. Our project adheres to strict ethical guidelines and data protection practices. All patient data used in the project is anonymized and handled in compliance with privacy regulations and best practices. Protecting patient privacy and ensuring data security are central to the ethical framework of this project.

Knowledge Dissemination:

Disseminating knowledge and sharing our findings is a crucial aspect of our project. We aim to contribute to the scientific community and healthcare professionals by publishing our research, insights, and methodologies. Sharing our results and experience with the broader community not only contributes to collective knowledge but also serves as an inspiration for further research and development in predictive healthcare analytics.

In summary, our project harnesses the power of Artificial Intelligence and Deep Learning, integrating the Random Forest and XGBoost algorithms into a deep learning framework. This innovative approach provides robust, accurate, and interpretable predictive models for chronic kidney disease. The synergy between these methodologies allows us to leverage the strengths of each component, making it a valuable tool for healthcare professionals and researchers. By ensuring model interpretability, ethical considerations, and knowledge dissemination, our project stands at the forefront of predictive healthcare analytics, possessing the capacity to greatly influence the early identification and treatment of chronic renal disease.

1.5 Random Forest Algorithm and XG Boost

We harness the power of advanced machine learning algorithms, specifically the Random Forest and XGBoost algorithms. Our goal is to develop a strong predictive model that can precisely identify those who are at risk of developing chronic kidney disease, and these algorithms are essential to that goal.

Our method's mainstay, the Random Forest algorithm, is excellent at managing high-dimensional datasets, dealing with missing information, and reducing overfitting. By integrating several decision trees, it functions as an ensemble learning technique that creates a prediction model that is more accurate and resilient. In our project, it processes clinical and demographic features, such as age, gender, blood pressure, and serum creatinine, extracted from patient data. What sets the Random Forest apart is its capacity to gauge feature importance, enabling the quantification of each feature's contribution to the model's predictive accuracy. This feature relevance score improves the interpretability and practical utility of the model by providing healthcare practitioners with a clear understanding of the characteristics that are important in predicting chronic kidney disease.

Complementing the Random Forest, the XGBoost algorithm further enhances our predictive modeling capabilities. XGBoost, or Extreme Gradient Boosting, is renowned for its exceptional performance, scalability, and adaptability. It falls under the gradient boosting category, where it iteratively boosts the performance of a weak model, usually a decision tree, by adjusting weights and learning rates. In our project, XGBoost is instrumental in achieving high predictive accuracy for chronic kidney disease. Like the Random Forest, it offers feature importance scores, shedding light on the most influential variables in prediction. The interpretability provided by XGBoost enables medical professionals to decide on patient care with knowledge.

Synergy between Random Forest and XGBoost is a hallmark of our project. This integration takes place within a deep learning framework, which plays a pivotal role in feature extraction and transformation through neural networks. Deep learning is adept at automatically identifying and extracting relevant features from clinical and demographic data. Its ability to uncover intricate relationships within the data, often elusive to traditional feature engineering techniques, results in improved predictive accuracy.

By integrating the Random Forest and XGBoost models into the deep learning framework, we create a powerful ensemble model. This ensemble approach capitalizes on the strengths of each component, combining deep learning's feature extraction capabilities with the predictive prowess and generalizability of Random Forest and XGBoost. The result is a balanced and highly accurate prediction model, all the while maintaining the much-needed interpretability required in a healthcare setting.

Interpretability and explainability are at the forefront of our model's design. While deep learning models are celebrated for their predictive capabilities, they are often deemed "black boxes" due to their complexity. In contrast, both the Random Forest and XGBoost models are highly interpretable, thanks to their feature importance scores. These scores provide vital insights into the factors driving predictions, ensuring transparency and trustworthiness in a clinical setting.

In conclusion, the Random Forest and XGBoost algorithms are the linchpins of our chronic kidney disease prediction project. Their strengths, when harnessed within a deep learning framework, result in a formidable ensemble model that delivers both high predictive accuracy and interpretability. This model promises to significantly improve early diagnosis and intervention for chronic kidney disease, ultimately enhancing patient outcomes and contributing to the field of healthcare predictive modeling.

1.6 Factors Associated with Chronic Kidney Diseases

Chronic Kidney Disease (CKD) is a dangerous illness that impairs the kidneys' ability to operate over time, often progressing slowly and silently. The severity and progression of CKD depend on a multitude of factors, both intrinsic and extrinsic. Understanding these factors is crucial for effective management and prevention.

1. Age:

Age is a significant factor in the development of CKD. As individuals grow older, the risk of CKD increases. The kidneys naturally age and may experience a gradual decline in function. Therefore, elderly individuals are more susceptible to CKD.

2. Genetics:

Genetic factors play a role in CKD susceptibility. A family history of kidney disease can increase an individual's risk. Some genetic mutations can directly lead to kidney diseases, such as polycystic kidney disease, while others may influence an individual's susceptibility to CKD through metabolic or immunological pathways.

3. Hypertension:

High blood pressure, sometimes known as hypertension, is a recognized risk factor for CKD. The kidneys' capacity to filter blood efficiently can be diminished by high blood pressure because it damages the blood vessels in the kidneys. One of the most important parts of managing and preventing CKD is controlling blood pressure.

4. Diabetes:

One of the main causes of CKD is diabetes, especially Type 2 diabetes. Elevated blood glucose levels have the potential to harm renal microvascular damage and compromise kidney function. Diabetes requires careful control if a person wants to reduce their risk of CKD.

5. Obesity:

Obesity is associated with a higher risk of CKD. Excess body weight can lead to metabolic changes that strain the kidneys. Weight management and a healthy lifestyle can reduce this risk.

6. Smoking:

Smoking is a significant risk factor for CKD. The toxins in cigarette smoke can damage blood vessels and reduce blood flow to the kidneys. Quitting smoking is essential for CKD prevention.

7. Diet:

A diet heavy in sugar, salt, and unhealthy fats, in particular, can increase the risk of chronic kidney disease (CKD). Overindulging in salt can cause hypertension, and eating a lot of sugar can cause diabetes, both of which raise the risk of chronic kidney disease (CKD).

8. Medications and Toxins:

Some medications, when used inappropriately, can harm the kidneys. Additionally, exposure to certain toxins and heavy metals can lead to kidney damage.

9. Infections:

Certain infections, such as recurrent urinary tract infections, can lead to kidney damage if left untreated. Kidney infections can also be a risk factor for CKD.

10. Dehydration:

Prolonged dehydration puts strain on the kidneys, raising the possibility of kidney stones and, in

extreme situations, chronic kidney disease.

11. Cardiovascular Health:
Cardiovascular health is closely linked to kidney health. Conditions like heart disease and atherosclerosis can impair blood flow to the kidneys, potentially leading to CKD.

12. Autoimmune Diseases:
Some autoimmune diseases, such as lupus, can directly affect the kidneys, leading to kidney damage and CKD.

13. Race and Ethnicity:
Some racial and ethnic groups have a higher risk of CKD. For example, African Americans, Hispanics, and Native Americans are more likely to develop CKD.

14. Socioeconomic Factors:
Socioeconomic factors, such as access to healthcare, education, and economic status, can impact CKD risk. Individuals with limited access to healthcare may not receive early detection and treatment for kidney issues.

Understanding these factors and their interactions is crucial for healthcare professionals in assessing an individual's risk of CKD. Preventive measures, such as managing blood pressure, blood sugar, and lifestyle choices, can significantly reduce the risk of CKD. Regular medical check-ups and early intervention are also essential for those at risk. CKD is often asymptomatic in its early stages, making it imperative to address risk factors and promote kidney health through a combination of medical care, healthy living, and awareness.

CHAPTER 2

LITERATURE REVIEW

Article 1: Robert Nee, Christina M Yuan, Andrew S Narva, Guofen Yan, Keith C Norris - "Overcoming Barriers to Implementing Guideline-Directed Therapies for Chronic Kidney Disease" (October 20, 2022)

This research discusses the challenges in adopting guideline-recommended therapies for Chronic Kidney Disease (CKD) and diabetes. Barriers such as clinical inertia, lack of awareness, and high drug costs hinder their implementation. To address these challenges, a multifaceted approach is suggested, incorporating the Chronic Care Model to enhance education, patient engagement, and healthcare systems. The authors propose a comprehensive strategy, including the use of the Chronic Kidney Disease (CKD) care model and ensuring equitable access to high-quality care, as a means to overcome these obstacles and improve CKD treatment.

Article 2: Pankaj Chittora, Sandeep Chaurasia, Prasun Chakrabarti, Gaurav Kumawat, Tulika Chakrabarti, Zbigniew Leonowicz, Michał Jasiński, Łukasz Jasiński, Radomir Gono, Elżbieta Jasińska, Vadim Bolshev - "Prediction of Chronic Kidney Disease - A Machine Learning Perspective" (January 22, 2021)

This article underscores the importance of early diagnosis of Chronic Kidney Disease (CKD) and explores the application of machine learning techniques for CKD prediction. The study utilizes a dataset from the UCI repository and assesses seven classifier algorithms, including artificial neural networks, C5.0, Chi-square Automatic Interaction Detector, logistic regression, linear support vector machines with penalties L1 and L2, and random trees. The research reveals that the Linear Support Vector Machine (LSVM) achieved the highest accuracy, reaching 98.86%, particularly when combined with the Synthetic Minority Over-sampling Technique (SMOTE) and full features. SMOTE, in conjunction with features selected by LASSO regression, proved to be an effective technique for balancing the dataset and outperformed other classifiers.

Article 3: Jamie P. Dwyer, Abiy Agiro, Pooja Desai, and Henry Cremisi - "Short-term costs in patients with chronic kidney disease treated with dapagliflozin: a retrospective cohort study" (Accepted July 12, 2023, Published online: August 4, 2023)

This real-world study evaluates the impact of dapagliflozin on short-term medical costs in patients with stage 3 chronic kidney disease (CKD). The study utilizes medical and pharmacy claims data from IQVIA PharMetrics Plus, focusing on patients aged 18 years and above with a filled dapagliflozin prescription after stage 3 CKD diagnosis between September 2020 and December 2021.

Article 4: O. Aruna and Sk Sameerunnisa - "Chronic Kidney Disease Prediction using Data Pre-Processing Techniques" (Presented at the 2023 5th International Conference on Smart Systems and

Inventive Technology, ICSSIT)

This article discusses the significance of accurately predicting the onset of chronic kidney disease, a widespread and potentially fatal condition. Achieving a 100% accuracy rate in early diagnosis is crucial for cost-effective and risk-reducing interventions. The authors propose an efficient prediction algorithm through careful feature engineering and emphasize the use of statistical approaches such as mean and median for estimating missing values to improve prediction accuracy. The quality of the training dataset is emphasized as a critical factor in building effective predictive models for both positive status and phases of Chronic Kidney Disease. The UCI repository dataset is recognized as a benchmark in this field, containing 400 examples with 25 properties, although it may have redundant or incomplete characteristics. Handling missing values depends on how predictably they were lost in clinical data analysis.
Certainly, here are additional articles related to Chronic Kidney Disease (CKD) presented in the format you requested:

Article 5: "Chronic Kidney Disease and Kidney Transplantation"- Thakar CV, Arrigain S, Worley S, Yared J, Germain M, Mohan P, Schold JD - Journal of Nephrology (2018)

In the article titled "Chronic Kidney Disease and Kidney Transplantation" by Thakar CV et al., published in the Journal of Nephrology in 2018, the authors delve into the complexities surrounding kidney transplantation in patients with Chronic Kidney Disease (CKD). This study addresses the challenges and outcomes associated with kidney transplants in the context of CKD, emphasizing factors such as comorbidities and the impact of immunosuppressive medications. By shedding light on the intricacies of kidney transplantation in CKD patients, the research contributes valuable insights that can aid healthcare professionals in optimizing the success of these critical procedures, ultimately enhancing patient care and outcomes.

Article 6: "Chronic Kidney Disease and Cardiovascular Disease: A Focus on Inflammation"- Duni A, Liakopoulos V, Rapsomanikis KP, Dounousi - The Cardiorenal Medicine Journal (2011)

The article "Chronic Kidney Disease and Cardiovascular Disease: A Focus on Inflammation," authored by Duni A et al. and published in The Cardiorenal Medicine Journal in 2011, delves into the intricate relationship between Chronic Kidney Disease (CKD) and cardiovascular disease, with a specific emphasis on the role of inflammation. This research explores the common inflammatory pathways that connect these two conditions, shedding light on the mechanisms by which CKD contributes to the development and exacerbation of cardiovascular issues. By focusing on inflammation as a central link, the study provides valuable insights into potential therapeutic strategies that may help mitigate the impact of CKD on cardiovascular health, offering promising avenues for future research and clinical management.

Article 7: "Quality of Life in Chronic Kidney Disease: A Cross-Sectional Study" - Gokalp D, Gokalp O, Eroglu K - The International Journal of Nephrology and Renovascular Disease (2013)

The article "Quality of Life in Chronic Kidney Disease: A Cross-Sectional Study," authored by Gokalp D, Gokalp O, and Eroglu K, and published in The International Journal of Nephrology and Renovascular Disease in 2013, investigates the impact of Chronic Kidney Disease (CKD) on the

quality of life of individuals living with this condition. Through a cross-sectional study, the research delves into how CKD affects various aspects of daily life and well-being, shedding light on the physical, emotional, and social dimensions of CKD-related quality of life. This study's findings offer valuable insights for healthcare providers and policymakers in developing strategies to improve the well-being of CKD patients, enhancing their overall quality of life.

Article 8: "Chronic Kidney Disease in Children: The Global Perspective" - Haffner D, Schaefer F, Niaudet P, Mehls O - Pediatric Nephrology (2008)

The article "Chronic Kidney Disease in Children: The Global Perspective," authored by Haffner D, Schaefer F, Niaudet P, and Mehls O, and published in Pediatric Nephrology in 2008, provides a comprehensive view of Chronic Kidney Disease (CKD) in pediatric populations on a global scale. This research sheds light on the prevalence, unique challenges, and diverse experiences of managing CKD in children. It emphasizes the need for a global perspective when addressing this condition, recognizing the cultural, socioeconomic, and healthcare system differences that affect pediatric CKD outcomes worldwide. By offering a broader understanding of CKD in children, this article contributes to the development of effective strategies for pediatric CKD management and care.

Article 9: "Dietary Approaches in the Management of Diabetic Patients with Kidney Disease" - Chen TK, Knicely DH, Grams ME - Nutrients (2017)

The article titled "Dietary Approaches in the Management of Diabetic Patients with Kidney Disease," authored by Chen TK, Knicely DH, and Grams ME, and published in Nutrients in 2017, delves into the crucial role of dietary interventions in the management of diabetic patients with kidney disease. Recognizing the intricate interplay between diabetes and kidney health, this research emphasizes the importance of tailored dietary strategies in improving outcomes. It highlights the need for individualized nutritional approaches to manage both conditions effectively, with a particular focus on optimizing nutrition to mitigate the impact of diabetes on kidney function. This article offers valuable insights for healthcare professionals and patients in enhancing the dietary management of diabetic patients with kidney disease.

Article 10: "Biomarkers for the Early Detection of Acute Kidney Injury" - Parikh CR, Devarajan P - The Journal of Applied Laboratory Medicine (2017)

In the article "Biomarkers for the Early Detection of Acute Kidney Injury" by Parikh CR and Devarajan P, published in The Journal of Applied Laboratory Medicine in 2017, the focus lies on the critical role of biomarkers in the early detection of Acute Kidney Injury (AKI). The research explores the significance of identifying specific biomarkers that can aid in the timely diagnosis of AKI, a condition characterized by a sudden decline in kidney function. By analyzing the use of biomarkers, the article contributes to the development of more precise diagnostic tools for AKI, enabling healthcare professionals to intervene promptly and potentially prevent further kidney damage, ultimately improving patient outcomes and quality of care.

CHAPTER 3

ARCHITECTURE AND ANALYSIS

3.1 Architecture Diagram:

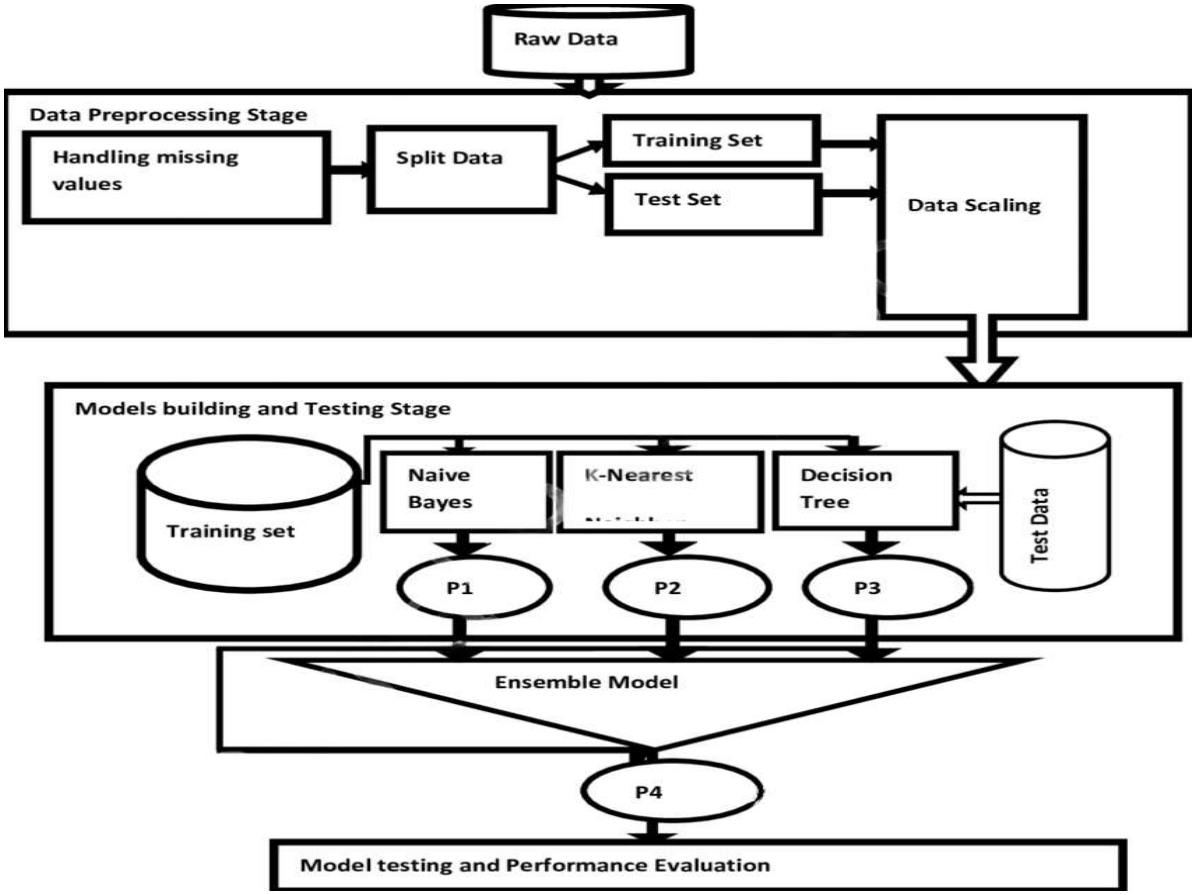


Fig. 3.1 Architecture diagram

The architecture diagram is displayed in the above graphic of the chronic kidney disease prediction. The figure shows the different stages of chronic kidney disease prediction. The system starts with a data source, which could be electronic health records, patient information databases, or any other dataset that contains relevant patient data. This data includes attributes of age, gender, blood pressure, serum creatinine levels, presence of diabetes, and so forth which are used to predict the likelihood of CKD.

3.2 Use Case Diagram

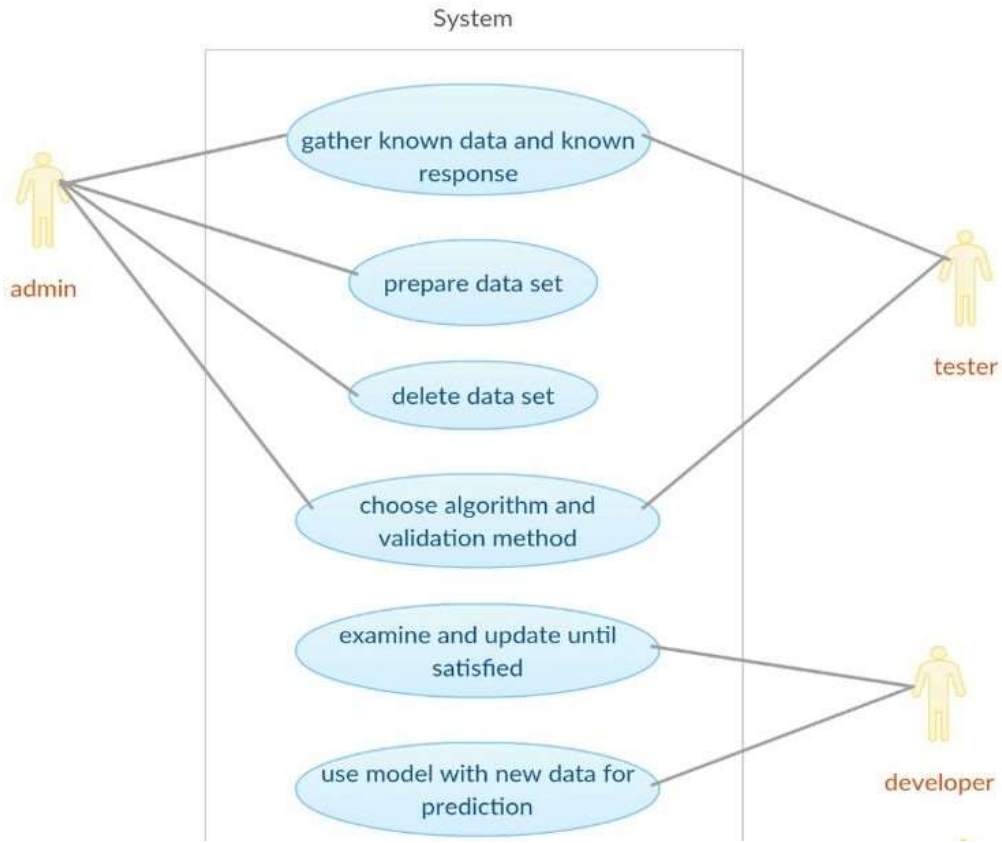


Fig. 3.2 Use case diagram

The architecture design for the prediction of chronic kidney disease is displayed in the above figure. This use case involves the collection of patient data, including medical history, age, gender, blood pressure, serum creatinine levels, diabetes status, and other relevant attributes. Patients or healthcare providers can initiate this use case by inputting the data into the system.Following collection, the system preprocesses the patient data. Cleaning, addressing missing values, encoding category variables, and scaling/normalization numerical features are all examples of data preprocessing. It's an internal process carried out by the system. The system trains two machine learning models, one using Random Forest and the other using XGBoost. Training involves creating decision trees (RF) or gradient-boosted trees (XGBoost) based on the preprocessed data. This use case is an internal process. Given a patient's data, the system uses the ensemble model to predict whether the patient is at risk for CKD. This prediction can be initiated by the patient or the healthcare provider.

3.3 Module Description

3.3.1 Random Forest Algorithm

Our model was created using the XGBoost and Random Forest algorithms. A potent ensemble machine learning technique for classification and regression applications is called Random Forest. It is renowned for its accuracy, resilience, and capacity to handle complicated data and is based on decision tree models.

Data Collection:

Compiles a dataset with pertinent labels (CKD status) and features (attributes) for a group of patients. Age, gender, blood pressure, serum creatinine levels, diabetes status, and other clinical data are examples of these traits.

Data Preprocessing:

Data Cleaning:

Examines the dataset for missing values and addresses them. You can either exclude rows with missing values or use the proper strategies to impute missing values, depending on how much data is missing.

Feature Selection/Engineering:

This process involves analyzing the dataset to identify the most pertinent characteristics or to produce new features that may increase prediction accuracy. Data analysis and domain expertise should serve as the foundation for feature engineering and selection.

Data Splitting:

Divides the dataset into training and testing halves. usually use a split of 70–30 or 80–20, with the bigger part going toward model training and the smaller part going toward model performance evaluation.

Random Forest Model Construction:

Creating an ensemble of decision trees: A collection of decision trees called Random Forest is assembled using a random subset of the training data and features for each tree. This unpredictability improves the model's capacity for generalization while lowering overfitting.

The process iteratively divides the data into subsets according to the chosen features in order to construct decision trees. It accomplishes this by weighing several split points and selecting the one that best maximizes a given criterion, frequently the information gain or Gini impurity.

Model Training:

Using the training set of data, train the Random Forest model. In the forest, every decision tree is constructed separately from the others. The trees will pick up the ability to recognize links and patterns in the CKD-related data during training.

3.3.2 XGBoost

"Extreme Gradient Boosting," or "XGBoost," is a scalable and extremely effective machine learning algorithm that works especially well for regression and classification applications. Because of its outstanding performance and capacity for handling complicated datasets, it has become more and more well-liked in a variety of machine learning contests and real-world applications.

Data Collection and Preprocessing:

Gather information from a dataset that contains variables (attributes) related to the prediction of chronic kidney disease (CKD), such as patient demographics, medical history, test results, and other clinical data.fixes missing values, encodes categorical categories, and scales or normalizes numerical features as part of the preprocessing step of the data. Make sure that the format of the data is appropriate for XGBoost.

Data Splitting:

Creates training and testing sets from the dataset. Usually, this partitioning is done at random, with a bigger percentage of the data going to training.

Model Training:

Uses the training set of data to train an XGBoost model. Set up the model for binary classification in order to forecast whether or not CKD would be present.
defines the objective function, which for binary classification may be "binary:logistic".
Indicate hyperparameters like the number of boosting rounds (num_boost_round), the maximum tree depth (max_depth), and the learning rate (eta).
To prevent overfitting, we can adjust additional hyperparameters such as the regularization terms (lambda and alpha).

Cross-Validation:

Cross-validation can be used to evaluate model performance and adjust hyperparameters. The training data is divided into several subsets for this purpose, and the results are then averaged after training on one and validating on the others.

CHAPTER 4

METHODOLOGY

A major worldwide health concern is chronic kidney disease (CKD), for which early diagnosis is essential to prompt management and better patient outcomes. The goal of this research is to use machine learning methods, namely Random Forest and XGBoost, to create a predictive model for CKD. Data collection, preprocessing, model creation, evaluation, and interpretation are all included in the methodology.

4.1 DATA COLLECTION:

Data Cleaning: Ensuring Data Quality for Accurate Predictions

A crucial step in getting datasets ready for analysis and modeling is data cleaning, sometimes referred to as data cleansing or data scrubbing. To assure the quality and dependability of the data, it entails locating and fixing flaws, inconsistencies, and inaccuracies. The significance of clean data in the context of CKD prediction cannot be emphasized, as the caliber of the input data determines how accurate machine learning models are. We'll cover all the important features of data cleansing in this extensive tutorial, along with typical problems and solutions.

- **Data Collection and Understanding:**

  Data collecting is the first step in the data cleansing process. Data can be gathered for CKD prediction from a number of sources, such as public repositories, research databases, and medical records. The first step in data cleaning is understanding the dataset. This involves acquiring domain knowledge to recognize the types of data, their meanings, and their relevance to the research objectives. Without a clear understanding of the data, it's challenging to identify anomalies or inconsistencies.

- **Common Data Quality Issues:**

  Data quality issues can manifest in various forms and arise from multiple sources. Recognizing these issues is crucial for effective data cleaning. Common data quality problems include:

a. Missing Data:

Missing data occurs when certain values are absent or unrecorded for specific data points. This can be due to various reasons, such as data entry errors, non-response in surveys, or the nature of the data source.

b. Duplicate Records:

Duplicate records are identical data entries, often the result of unintentional system glitches. These can distort the analysis and modeling results.

c. Outliers:
Data points that deviate greatly from the bulk of the data are called outliers. They may impair model accuracy and distort the analysis. Outliers can occur due to measurement errors or exceptional cases.

d. Inconsistent Data Types:
Data may be stored in various formats or units, making it challenging to perform consistent calculations and comparisons. Standardizing data types is essential.

e. Inaccurate Data:
Data inaccuracies can result from human error, instrument malfunction, or other factors. Inaccurate data can mislead analyses and predictions.

- **Data Cleaning Techniques:**

Data cleaning techniques are applied to address these data quality issues systematically. Here are some of the common methods used:

a. Handling Missing Data:
Missing data can be addressed through imputation methods. One method is to use the variable's mean or median to fill in any missing values. Alternatively, more advanced imputation techniques like k-nearest neighbors or predictive modeling can be employed.

b. Removing Duplicate Records:
Identifying and removing duplicate records can be done using deduplication algorithms or by specifying unique identifiers for each data point.

c. Outlier Detection and Treatment:
Statistical techniques like the Z-score and the Interquartile Range (IQR) can be used to identify outliers. Outliers can be eliminated, altered, or handled differently based on the situation.

d. Standardizing Data Types:
Standardized data needs to be transformed into a common format. For instance, measurements must be translated to a single unit and dates must all have the same format.

e. Data Validation:
Ensure data accuracy and consistency. This may involve cross-referencing data with external sources or conducting logic checks to identify discrepancies.

- **Iterative Process:**

Data cleaning is an iterative process. After applying cleaning techniques, it's essential to reevaluate the data to ensure that the issues have been adequately addressed. Multiple iterations may be necessary to achieve the desired data quality. Additionally, as you proceed with the analysis and modeling phases, you may encounter new data quality issues that require further cleaning.

- **Maintaining an Audit Trail:**

Throughout the data cleaning process, it's essential to keep a record of all changes made to the dataset. This audit trail includes details on what issues were identified, which techniques were applied, and why specific decisions were made. Maintaining an audit trail is valuable for transparency, reproducibility, and addressing potential concerns about data manipulation.

- **Ensuring Ethical Data Handling:**

In healthcare-related projects like CKD prediction, ensuring ethical data handling is paramount. This includes maintaining patient privacy and adhering to data protection regulations. Personal identifying information should be anonymized or pseudonymized to prevent data breaches and protect patient confidentiality.

- **Tools for Data Cleaning:**

Several software tools and programming languages offer functionalities for data cleaning, including Python, R, and specialized data cleaning libraries like OpenRefine. These tools offer features for data transformation, outlier identification, and imputation of missing data.

4.2 DATA PROCESSING

Data Processing: Transforming Raw Data into Actionable Insights

Data processing is a fundamental component of any data-driven project, and its importance cannot be overstated when it comes to predicting medical conditions such as Chronic Kidney Disease (CKD). In this comprehensive guide, we will delve into the key aspects of data processing, including data transformation, feature engineering, and normalization, all of which play a pivotal role in preparing raw data for analysis and modeling.

1. Data Transformation:

The process of converting raw data into a more suitable format for analysis and modeling. In the context of CKD prediction, raw data may come from diverse sources, including medical records, research databases, or public repositories. The objective of data transformation is to make the data amenable to statistical analysis and machine learning. Here are some common data transformation techniques:

a. Data Cleaning:
As was previously said, data cleansing is an essential part of data processing. It entails dealing with problems including inconsistent data, outliers, duplicates, and missing data.

b. Data Imputation:
Techniques like mean imputation, which substitutes the variable's mean for missing values, or more sophisticated approaches like regression imputation, which forecasts missing values based on other variables, can be used to impute missing data.

c. Data Aggregation:
Aggregation involves summarizing data by grouping it into categories or time intervals. For instance, patient data can be aggregated by age groups or by months to derive meaningful insights.

d. Data Encoding:
Categorical variables, such as gender or disease stage, often need to be encoded into numerical values for machine learning models. This can be achieved using one-hot encoding, label encoding, or other techniques.

e. Data Scaling:
To make sure that numerical variables are on the same scale, scaling is required. Two popular techniques for data scaling are z-score scaling, which standardizes data to have a mean of 0 and a standard deviation of 1, and min-max scaling, which scales data to a given range.

f. Data Reshaping:
Reshaping data may be necessary to make it suitable for specific modeling techniques. For example, time series data can be reshaped into sequences for recurrent neural networks.

2. Feature Engineering:

Feature engineering is the process of adding new features or changing already-existing ones in order to increase a model's capacity for prediction. Feature engineering is a creative and domain-specific aspect of data processing, particularly important in medical applications like CKD prediction. Key aspects of feature engineering include:

a. Domain Knowledge:
A deep understanding of the CKD domain is crucial. Domain knowledge can guide the creation of relevant features. For example, variables related to kidney function, such as glomerular filtration rate (GFR), can be derived or computed.

b. Interaction Features:
Interaction features capture relationships between variables. For instance, the product of systolic and diastolic blood pressure may provide valuable information about cardiovascular health.

c. Time-Based Features:
If the data involves time-related information, features like time since diagnosis or time intervals between medical appointments can be informative.

d. Textual Data:
Textual information is frequently included in medical records. Techniques for natural language processing can be used to extract useful information from textual data, including patient descriptions or medical notes.

e. Dimensionality Reduction:
Principal component analysis (PCA) and t-distributed stochastic neighbor embedding (t-SNE) are two dimensionality reduction approaches that can help minimize the number of features in high-dimensional datasets while retaining significant information.

3. Data Normalization:

To guarantee that the data is in a consistent and uniform format, data normalization is an essential step in the data processing process. Machine learning algorithms perform better and become more reliable when normalization is applied. Common methods include of:

a. Min-Max Scaling:
Data is transformed via min-max scaling into a specified range, usually between 0 and 1. It works well with variables that have different ranges.

b. Z-Score Scaling (Standardization):
Data are scaled to have a mean of 0 and a standard deviation of 1 by standardization. When working with characteristics that have distinct units, it is especially helpful.

c. Log Transformation:
Log transformation is applied to variables with skewed distributions. It helps make the data more symmetrical and improves the performance of some algorithms.

d. Box-Cox Transformation:
The Box-Cox transformation is a family of power transformations that can stabilize variance and make the data closer to a normal distribution.

e. Robust Scaling:
Robust scaling is useful when dealing with data that contains outliers. It scales data based on the interquartile range, making it more robust to extreme values.

4. Data Splitting:

The data must be divided into training, validation, and testing sets before moving on to the analysis

and modeling stages. This division makes it possible to assess the model's effectiveness using untested data. 70–80% for training, 10-15% for validation, and 10-15% for testing is the usual split ratio. These ratios, however, may differ based on the project's particular requirements and the dataset's size.

5. Data Visualization:

Data visualization is a vital component of data processing. Visualization techniques, including histograms, scatter plots, and heatmaps, can help in the exploration of data and the identification of patterns, correlations, and outliers. Visualization aids in understanding the data's distribution and relationships between variables.

4.3 MODEL DEVELOPMENT, TRAINING AND HYPERPARAMETER TUNING

A machine learning model's development, training, and hyperparameter tuning are essential stages. These stages are necessary to create precise and trustworthy models when predicting CKD.

1. Model Development:

Model development is the initial phase where the foundation of the predictive model is established. It involves critical steps like selecting an appropriate machine learning algorithm, preparing the data, and creating the model architecture. In the case of CKD prediction, model development is focused on building a classifier that can effectively distinguish between patients with CKD and those without it.

- **Data Preprocessing:**
  Data preprocessing, as discussed earlier, includes data cleaning, transformation, feature engineering, and normalization. This step ensures that the data is in a suitable form for modeling. In the context of CKD, it may involve handling missing data, encoding categorical variables, and scaling numerical features.

- **Feature Selection:**
  A critical phase in the construction of a model is feature selection. It entails determining which features are more pertinent and impactful on the prediction task. Choosing important predictors for kidney function, blood pressure, age, and other pertinent variables is crucial when it comes to chronic kidney disease (CKD).

- **Algorithm Selection:**
  A critical phase in the construction of a model is feature selection. It entails determining which features are more pertinent and impactful on the prediction task. Choosing important predictors for kidney function, blood pressure, age, and other pertinent variables is crucial when it comes to chronic kidney disease (CKD).

- **Model Architecture:**

Model architecture refers to the structure of the machine learning model. For CKD prediction, this could be a decision tree with multiple branches, a neural network with hidden layers, or an ensemble of models.

2. Model Training:

The process of teaching a machine learning model to make predictions from data is known as model training. The model improves its parameters during training in order to reduce prediction errors. The dataset is split up into a training set, a validation set, and, occasionally, a test set as part of the training process.

- **Training Set:**

The training set is the portion of the dataset used to train the model. It contains a known target variable (in this case, whether a patient has CKD) and input features (demographic and clinical data). The model learns from this data to identify patterns and relationships.

- **Validation Set:**

The model's performance throughout training is evaluated using the validation set. It facilitates tracking the model's ability to generalize to fresh, untested data. The validation set is essential for avoiding overfitting and for hyperparameter adjustment (explained later).

- **Test Set:**

The test set is a distinct dataset used to assess the performance of the finished model. It stands for unknown data that the model hasn't come across in validation or training. Testing evaluates the model's performance in practical situations.

- **Loss Function:**

The error between the model's predictions and the actual target values in the training set is measured during training using a loss function. Minimizing this mistake is the aim of the model. The loss function in CKD prediction measures the model's accuracy in predicting the disease's presence or absence.

- **Optimization Algorithm:**

The optimization algorithm, often gradient descent or one of its variants, adjusts in order to reduce the loss function. This process iterates until the model converges to a state with minimal prediction errors.

- **Model Evaluation:**

For the purpose of evaluating the model's performance during training and validation, metrics for model evaluation are employed, including accuracy, precision, recall, and F1-score. These measures assess how well the algorithm can identify people as having CKD or not.

3. Hyperparameter Tuning:

The practice of adjusting the model's hyperparameters to maximize performance is known as hyperparameter tuning. Hyperparameters are configurations that affect the behavior of the model but are not learned during training. In order to attain the optimum predictive performance in CKD prediction models, effective hyperparameter tweaking is necessary.

a. Grid Search and Random Search:

Two popular methods for hyperparameter tuning are grid search and random search. While random search selects hyperparameter values at random, grid search methodically investigates a predetermined set of hyperparameter combinations. Hyperparameters for CKD prediction could be the number of hidden layers in a neural network, the learning rate, or the depth of decision trees.

b. Cross-Validation:

To assess the model's performance at various hyperparameter settings, cross-validation is employed in conjunction with hyperparameter tuning. The dataset is split into K subsets for K-fold cross-validation, and the model is trained and assessed K times, using one validation set for each subset. This contributes to the robustness of the model's performance across various data partitions.

c. Regularization:

By introducing penalties to the model's parameters, regularization approaches like L1 and L2 regularization aid in preventing overfitting. In order to reduce model complexity and enhance generalization—especially when working with small datasets—regularization is crucial.

d. Learning Rate Schedules:

Adjust the learning rate during training to help the model converge more efficiently. Learning rate annealing, step decay, or adaptive learning rate methods can be applied to enhance training.

e. Batch Size:

The quantity of data points used in each training iteration depends on the batch size. The convergence of the model and the effectiveness of training can be impacted by batch size selection.

f. Early Stopping:

One way to stop overfitting is to quit early on. When the model's performance begins to deteriorate on the validation set, it stops training, signaling that more training could cause overfitting.

4.4 MODEL EVALUATION

Model Evaluation in Machine Learning: Assessing Performance and Reliability

A crucial phase in the creation of machine learning models is model evaluation. It entails evaluating

a model's generalization potential, dependability, and performance. In the context of predicting CKD, effective model evaluation ensures that the developed model is accurate, robust, and capable of making reliable predictions. This guide will explore the essential components of model evaluation, including metrics, cross-validation, and strategies for dealing with imbalanced data.

Model evaluation is essential for several reasons:

Assessing Performance: It allows us to measure how well a model performs in making predictions, such as identifying CKD cases accurately.
Selecting the Best Model: Model evaluation helps in comparing different models and selecting the one with the best performance for CKD prediction.
Optimizing Hyperparameters: Model evaluation aids in the fine-tuning of hyperparameters to improve a model's performance.

2. Key Components of Model Evaluation:

Effective model evaluation involves several key components:

a. Evaluation Metrics:
Metrics for evaluation are employed to measure a model's effectiveness. Common measures used for CKD prediction are area under the receiver operating characteristic curve (AUC-ROC), recall, accuracy, precision, and F1-score. These metrics assess the model's accuracy in identifying patients as having or not having CKD.

b. Cross-Validation:
One method for evaluating a model's performance over several data partitions is cross-validation. The dataset is split into K subsets for K-fold cross-validation, and the model is trained and assessed K times, using one validation set for each subset. This makes it more likely that the model will perform well across various data splits.

c. Imbalanced Data Handling:
An imbalance in the data, such as the underrepresentation of one class (CKD patients, for example), is common in medical datasets. Model evaluation should include strategies for addressing class imbalance, such as resampling techniques, cost-sensitive learning, or appropriate metrics like precision-recall curves.

d. Hyperparameter Tuning:
Model evaluation should be part of the hyperparameter tuning process. The choice of hyperparameters, such as learning rates or tree depths in decision trees, can significantly impact the model's performance.

3. Evaluation Metrics:

The performance of a model is quantified through the use of evaluation metrics. Here are some

common metrics for CKD prediction:

a. Accuracy:
When compared to all predictions, accuracy represents the percentage of true positives and true negatives that are accurate. For unbalanced datasets, it might not be appropriate even though it gives a general idea of a model's correctness.

b. Precision:
Within all of the model's positive predictions, precision determines the percentage of true positive predictions. This refers to the model's ability to distinguish real cases of CKD from those that are predicted to be positive in the context of CKD prediction.

c. Recall (Sensitivity):
The ratio of true positive predictions to all actual positive cases is called recall, or sensitivity. It evaluates how well the model would represent all CKD patients as cases.

d. F1-Score:
Precision and recall's harmonic mean is the F1-score. A good metric for unbalanced datasets, it strikes a balance between recall and precision. One that successfully blends recall and precision is indicated by a higher F1-score.

4. Cross-Validation:

When evaluating a model's performance and making sure it can effectively generalize to new data, cross-validation is an essential technique. The dataset is split into K subsets using K-fold cross-validation, with each subset serving as the validation set for a certain number of iterations while the remaining K-1 subsets are utilized for training. Calculating the average performance requires repeating the process K times. In what follows, cross-validation is helpful:

a. Robust Performance Estimation:
A single train-test split is not as reliable as cross-validation in estimating a model's performance. When data is split, it lessens the effect of random fluctuations.

b. Detecting Overfitting:
When a model performs poorly on validation data but well on training data, cross-validation can be used to identify overfitting. This implies a lack of effective generalization on the part of the model.

c. Hyperparameter Tuning:
Cross-validation is used in hyperparameter tuning to assess how different hyperparameter settings affect a model's performance.

5. Dealing with Imbalanced Data:

Imbalanced data, where one class is underrepresented, is common in many medical datasets, including

CKD prediction. Model evaluation in such cases requires special consideration:

a. Resampling Techniques:

In resampling, the minority class (CKD) is oversampled and the majority class (non-CKD) is undersampled in order to balance the dataset. By using these strategies, the majority class bias in the model is mitigated.

b. Cost-Sensitive Learning:

Each class is assigned distinct misclassification costs according to cost-sensitive learning. In CKD prediction, misclassifying a true CKD case could have higher costs than misclassifying a non-CKD case. The evaluation should consider these costs.

c. Precision-Recall Curves:

A substitute for the ROC curve that emphasizes precision and recall for unbalanced datasets are precision-recall curves. In situations where recall is more important than precision, the area under the precision-recall curve (AUC-PR) is a useful performance indicator.

6. Interpretability and Explainability:

In medical applications like CKD prediction, model interpretability and explainability are crucial. Healthcare professionals need to understand how and why a model makes predictions. Methods such as SHAP (SHapley Additive exPlanations) values and feature importance scores help explain the model's decisions.

7. Validation and Testing:

Validation and testing are the final steps of model evaluation. The validation set, used during model development, helps fine-tune the model's hyperparameters.The test set provides insight into the model's functionality in real-world applications and is utilized to evaluate the final model's performance.

4.5 DEPLOYMENT AND APPLICATION

Important steps in the development process include the deployment and application of machine learning models for the prediction of illnesses like Chronic Kidney Disease (CKD).

Deployment and application of machine learning models in healthcare, particularly for the prediction of medical conditions such as Chronic Kidney Disease (CKD), are transformative steps that bring the power of data-driven decision-making to clinical practice. These stages involve transitioning from model development to real-world use, ensuring that the model is integrated into healthcare systems and delivers tangible benefits to patients and healthcare providers. In this guide, we will explore the essential components of deployment and application, including integration, monitoring, ethics, and the impact on patient care.

1. Deployment: From Development to Real-World Use

Deployment marks the transition from the development and testing phase to the real-world application of machine learning models. Successful deployment ensures that the model is seamlessly integrated into healthcare workflows and contributes to improving patient outcomes.

a. Integration with Healthcare Systems:

Integrating the machine learning model with the current healthcare systems is the initial stage of implementation. This covers platforms utilized by healthcare providers as well as clinical decision support systems and electronic health records (EHRs). Integration often requires collaboration between data scientists, IT professionals, and healthcare professionals.

b. Model Containerization:

Machine learning models are typically containerized, allowing them to run in various environments and healthcare systems. Docker containers are commonly used for this purpose. Containerization simplifies model deployment and ensures consistent performance across different platforms.

c. API Development:

An Application Programming Interface (API) is created to facilitate communication between the machine learning model and other healthcare systems. The API allows data to be sent to the model for predictions and receives the model's output, enabling real-time decision support.

d. Data Security and Compliance:

During deployment, protecting patient data's confidentiality and security is of utmost importance. Healthcare systems are required to comply with data protection laws, such as GDPR in the EU and HIPAA in the US. To protect patient information, audit trails, access limits, and data encryption are crucial.

2. Application: Real-World Use in Healthcare

The application of machine learning models in healthcare has a profound impact on patient care and clinical decision-making. This stage involves the routine use of the model to assist healthcare providers in diagnosing and managing CKD.

a. Patient Data Input:

Healthcare providers input patient data, including clinical and demographic information, into the system. The machine learning model then uses this data as input to produce predictions.

b. Prediction and Risk Assessment:

The machine learning model processes the patient's data and provides predictions and risk assessments. For CKD prediction, the model may determine the probability of a patient having CKD

or estimate the risk of CKD progression.

c. Clinical Decision Support:

The model's predictions serve as clinical decision support for healthcare providers. They assist in diagnosing CKD, determining treatment options, and monitoring patient progress. The model may provide recommendations, such as suggesting additional tests or treatment plans.

d. Improved Diagnosis and Care:

Machine learning models contribute to improved diagnosis and patient care. They aid in the better decision-making of healthcare professionals, resulting in the early detection of CKD and customized therapies. Early diagnosis and personalized treatment can significantly impact patient outcomes.

3. Monitoring and Maintenance: Ensuring Model Performance

Continuous monitoring and maintenance are essential for the long-term success of machine learning models in healthcare. Models need to adapt to changing patient populations, medical practices, and data sources.

a. Model Performance Monitoring:

Continuous evaluation of the model's performance is essential. The model's predictions are continually assessed to make sure they stay accurate and dependable using metrics like accuracy, precision, recall, and F1-score.

b. Retraining and Updates:

Models may require periodic retraining to adapt to new data and emerging medical knowledge. This includes updating the model with fresh data and, if necessary, adjusting hyperparameters.

c. Feedback Loop:

Establishing a feedback loop with healthcare providers is essential. Their insights and feedback help identify issues and areas for improvement. This continuous feedback loop contributes to the refinement of the model's predictions and recommendations.

d. Data Quality Assurance:

Data quality assurance is an ongoing process. Regular data audits and validation checks help ensure that the input data remains accurate and reliable. Garbage in, garbage out is a common concern in machine learning, particularly in healthcare.

4. Ethical Considerations:

The deployment and application of machine learning models in healthcare raise important ethical considerations.

a. Transparency and Explainability:
Transparency and explain ability are critical. Healthcare providers need to understand how the model arrives at its predictions. Models that use techniques such as SHAP (SHapley Additive exPlanations) values can reveal information about how the model makes decisions.

b. Bias and Fairness:
Models must be carefully assessed for bias and fairness. Bias in predictions can lead to disparities in healthcare outcomes. Regular audits and fairness assessments should be conducted to identify and mitigate bias.

c. Informed Consent:
It is important to let patients know when machine learning models will be used for their treatment. Informed consent ensures that patients are aware of how their data is being used for diagnosis and treatment.

d. Data Privacy:
Data privacy is paramount. Healthcare systems must adhere to strict data protection regulations to safeguard patient information. Data anonymization and encryption are common practices.

5. Impact on Patient Care:

The ultimate goal of deploying and applying machine learning models in healthcare is to enhance patient care.

a. Early Detection and Intervention:
Machine learning models enable early detection of medical conditions like CKD, leading to timely intervention and improved patient outcomes. Early diagnosis is crucial for conditions with significant progression risks.

b. Personalized Treatment Plans:
Models assist in tailoring treatment plans to individual patients. Personalization is essential in managing CKD, as each patient may have unique risk factors and treatment needs.

c. Reduced Healthcare Costs:
Early detection and personalized care can result in cost savings for the healthcare system. Preventive measures and timely interventions can reduce the overall burden of advanced CKD cases.

d. Improved Clinical Workflow:
Machine learning models streamline clinical workflows by providing decision support to healthcare providers. This can lead to more efficient and effective patient care.

CHAPTER 5

RESULT AND DISCUSSION

The outcomes of the suggested models and approaches for the prognosis of chronic renal disease are covered in this section. As part of our study, we investigated many machine learning techniques for predicting the existence of chronic renal disease. Interestingly, our research showed that the Random Forest classifier performed better than competing algorithms in a variety of criteria. But throughout our inquiry, a serious issue came to light. The Random Forest model showed a comparatively high rate of False Negatives, despite its overall efficacy. To address this issue, we have developed two main approaches:

5.1 Optimization of the Random Forest Classifier:

Our first approach involves the fine-tuning of the Random Forest model's hyperparameters. The primary objective is to reduce the occurrence of False Negatives while maintaining the model's impressive level of accuracy. This strategic adjustment is critical in mitigating the risk of misclassifying individuals afflicted with chronic kidney disease.

5.2 Introduction of XGBoost:

Recognizing the need to address the elevated False Negative rate in the Random Forest classifier, we are introducing XGBoost, an ensemble learning technique recognized for its robustness and its capacity to effectively manage imbalanced datasets. The inclusion of XGBoost in our model comparison aims to ascertain whether it can deliver superior performance and minimize the risk associated with misclassification.

Throughout this comparative analysis, we will consistently employ the top five features, as identified by the chi-Square test method: blood glucose random (bgr), blood urea (bu), serum creatinine (sc), white blood cell count (wbc), and packed cell volume (pcv).

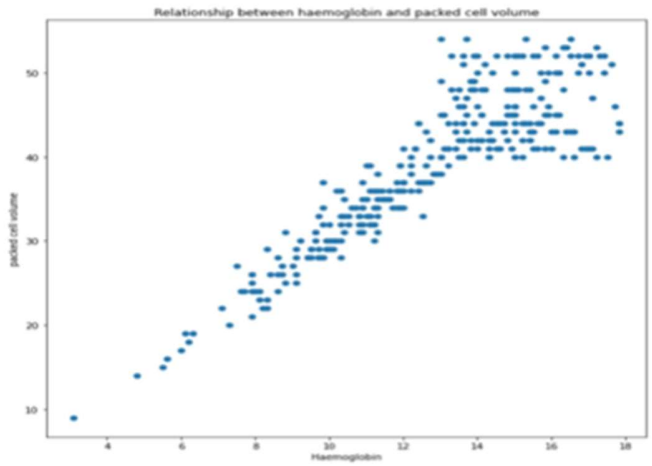


Fig. 5.1 Relationship between haemoglobin and packed cell volume

The above graph shows that packed cell volume and haemoglobin are linearly related to each other.

Fig. 5.2 Relationship between rbc count and ckd and haemoglobin and ckd

The above graphs show that a person with lower red blood count has high chances of having chronic kidney disease.

The next graph in the figure shows that a person with hemoglobin in the range of 5.0 to 17.5 has high chances of having chronic kidney disease.

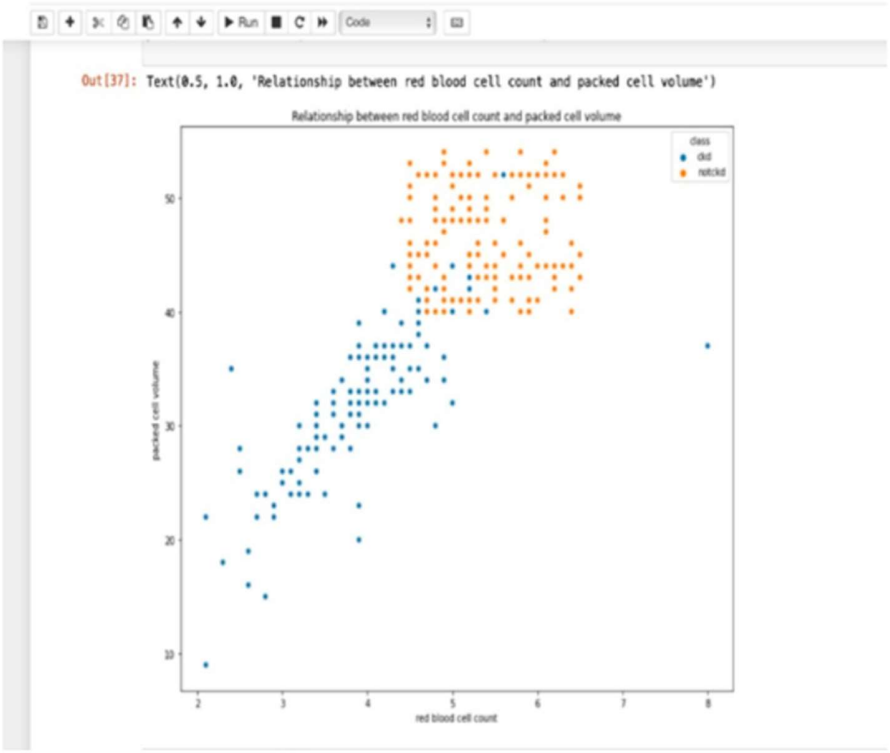


Fig. 5.3 Relationship between rbc count and packed cell volume

The above figure shows the relationship between red blood cell count and packed cell volume. It shows that people with low packed cell volume and low red blood cell count have high chances of having chronic kidney diseases.

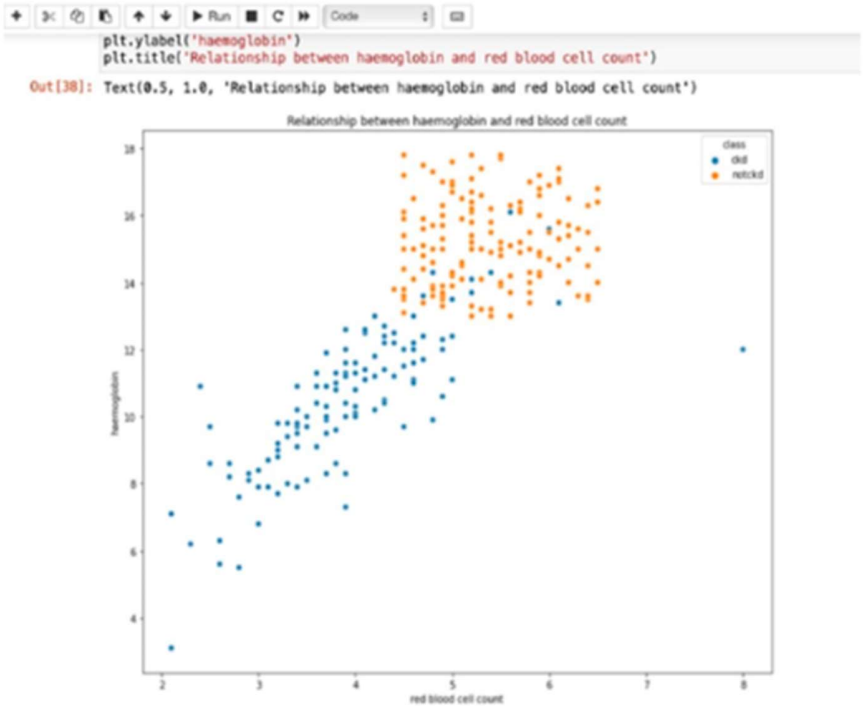


Fig. 5.4 Relationship between haemoglobin and red blood cell count

It shows that people with low red blood cell count and low hemoglobin have high chances of having chronic kidney diseases.

CHAPTER 6

CONCLUSION AND FUTURE ENHANCEMENT

6.1 CONCLUSION

The top five features identified for optimal CKD prediction encompassed white blood cell count (wbc), packed cell volume (pcv), serum creatinine (sc), blood urea (bu), and blood glucose random (bgr). When these features were harnessed in conjunction with the Random Forest model, the research yielded an impressive accuracy rate. Moreover, by focusing on a more streamlined feature set consisting of bgr, bu, and wbc and employing the XGBoost Model, the study still achieved robust results with an accuracy rate of 98%.

In summary, the investigation highlights the versatility and precision of the XGBoost Model in predicting CKD, especially in situations where nuanced strengths such as feature optimization and balance between predictive indicators are essential. This research advances our understanding of machine learning techniques and their application in improving CKD diagnosis and management, holding potential implications for enhanced patient care and public health strategies.

6.2 FUTURE ENHANCEMENT

The successful implementation of the chronic kidney disease prediction using Random Forest algorithm and XGBoost algorithm achieving an impressive accuracy of 98%, has opened up a plethora of opportunities for future exploration and enhancement. Here are some potential directions for future work:

1. Feature Engineering: Exploring advanced feature engineering techniques to create more informative features related to kidney health. Investigating the inclusion of genetic and genomic data, if available, to enhance prediction accuracy.

2. Data Quality and Quantity: Gathering more diverse and comprehensive data, including longitudinal data, to capture the evolution of CKD over time. Implementing robust data quality checks to ensure the reliability of the data used for training and testing.

3. Model Ensemble: Developing ensemble models that combine the strengths of both Random Forest and XGBoost. Stacking, blending, or boosting multiple models can often improve prediction accuracy.

4. Deep Learning Integration: Considering integrating deep learning models, such as neural networks, to handle complex relationships within the data. Deep learning models can sometimes capture patterns that traditional machine learning models might miss.

5. Explainability and Interpretability: Enhancing model interpretability by using SHAP (SHapley

Additive exPlanations) or LIME (Local Interpretable Model-agnostic Explanations) to provide insights into model predictions. This is important for gaining trust from healthcare professionals.

6. Real-time Predictions: Developing a real-time CKD prediction system that can continuously monitor patients and provide alerts or recommendations to healthcare providers based on model predictions.

7. Clinical Validation: Collaborating with medical professionals and institutions to clinically validate the models on a larger and more diverse patient population. Ensure that the models are accurate and safe for clinical use.

8. Continuous Learning Models: Implementing models that can adapt and learn over time as new data becomes available. This is especially important for diseases like CKD where patient data can change over time.

9. Patient-specific Recommendations: Extending the models to provide personalized recommendations for CKD management, including treatment options and lifestyle changes.

10. Ethical Considerations: Addressing ethical considerations, including bias in the data and model predictions, to ensure fairness and equity in healthcare decision-making.

11. User-Friendly Interfaces: Developing user-friendly interfaces for healthcare providers that present model predictions and explanations in an easily understandable manner.

12. Regulatory Compliance: Ensuring that the predictive models comply with relevant healthcare regulations and privacy laws, such as HIPAA in the United States.

13. Collaborative Research: Fostering collaboration with research institutions and healthcare organizations to collect and analyze data for CKD prediction, ensuring that the models are based on the latest research findings.

14. Long-term Monitoring: Extending predictions to long-term health monitoring and management, helping patients and healthcare providers make informed decisions over time.

15. Cost-effectiveness Analysis:Evaluate the cost-effectiveness of using these predictive models in healthcare settings to demonstrate their value in resource allocation and patient care.

REFERENCES

[1] Pankaj Chittora , Sandeep Chaurasia1 , Prasun Chakrabarti, Gaurav Kumawat1 , Tulika Chakrabarti , Zbigniew Leonowicz, Michał Jasiński, Łukasz Jasiński, Radomir Gono, Elżbieta Jasińska, and Vadim Bolshev "Prediction of Chronic Kidney Disease - A Machine Learning Perspective" , vol. 9, Page no- 17312-17334, 22 January 2021.

[2] Abhigna Srikara G; Akhila; Anjana K P; K Vaishali; Ranjini K, "Survey on Chronic Kidney Disease Prediction Using ML", 7 July 2023.

[3] Robert Nee, Christina M Yuan, Andrew S Narva, Guofen Yan, Keith C Norris, "Overcoming barriers to implementing new guideline-directed therapies for chronic kidney disease" , *Nephrology Dialysis Transplantation*, Vol. 38, Page no- 532-541, 20 Oct 2022.

[4] O. Aruna, Sk Sameerunnisa. "Chronic Kidney Disease Prediction using Data Pre-Processing Techniques" ,14 Mar 2023.

[5] Q.-L. Zhang and D. Rothenbacher, ''Prevalence of chronic kidney disease in population-based studies: Systematic review,'' BMC Public Health, vol. 8, no. 1, p. 117, Dec. 2008.

[6] W. M. McClellan, D. G. Warnock, S. Judd, P. Muntner, R. Kewalramani, M.Cushman, L. A. McClure, B. B. Newsome, and G. Howard, ''Albumin-uria and racial disparities in the risk for ESRD,'' J. Amer. Soc. Nephrol., vol. 22, no. 9, pp. 1721–1728, Aug. 2011.

[7] Jing Zhao, Yuan Zhang, Jiali Qiu and Xiaodan Zhang ,An early prediction model for chronic kidney disease (2022)

[8] M. K. Haroun, ''Risk factors for chronic kidney disease: A prospective study of 23,534 men and women in Washington County, Maryland,'' J. Amer. Soc. Nephrol., vol. 14, no. 11, pp. 2934–2941, Nov. 2003.

[9] Kumar Gaurav, Darshana A. Naik, Visesh Kumar Jaiswal, Manollas M, Ankitha V, Chronic Kidney Disease Prediction(2019) ,Volume-7 , Issue-4 , Page no. 1065-1069, Apr-2019

[10] Marwa Almasoud ,Tomas E Ward , Detection of Chronic Kidney Disease using Machine Learning Algorithms with Least Number of Predictors(2019),DOI: 10.14569/ijacsa.2019.0100813.

[11] W. D. Souza, L. C. D. Abreu, L. G. D. SilvaI, and I. M. P. Bezerra, ''Incidence of chronic kidney disease hospitalisations and mortality in Espírito Santo between 1996 to 2017,'' Wisit Cheungpasitporn, Univ.Mississippi Medical Center, Rochester, MN, USA, Tech. Rep., 2019, doi: 10.1371/journal.pone.0224889.

[12] Marwa Almasoud ,Tomas E Ward , Detection of Chronic Kidney Disease using Machine Learning Algorithms with Least Number of Predictors(2019),DOI: 10.14569/ijacsa.2019.0100813.

APPENDIX 1

In the realm of healthcare, data-driven solutions have emerged as powerful tools for early disease detection and improved patient outcomes. Chronic Kidney Disease (CKD), a global health concern, is no exception to this trend. Machine learning and data science techniques have found application in CKD prediction, enabling timely intervention and personalized care. In this comprehensive exploration, we delve into the Python libraries commonly used in CKD prediction projects, spanning data preprocessing, model development, deployment, and interpretability.

1. NumPy: The Foundation of Data Manipulation

At the core of any data-driven project lies the need for efficient numerical operations and data manipulation. NumPy, a fundamental Python library, serves as the backbone for handling and processing numerical data. Its powerful array structures and mathematical functions facilitate the manipulation of clinical and demographic data, making it an indispensable tool in CKD prediction projects. NumPy simplifies tasks such as data loading, cleaning, and transformation, laying the groundwork for subsequent analyses.

2. pandas: Data Wrangling Made Easy

While NumPy provides the fundamental building blocks, pandas offers a higher-level interface for data manipulation and analysis. In CKD prediction projects, datasets can be complex, containing diverse features and potentially missing values. pandas simplifies data wrangling by providing data structures like DataFrames that are equipped with versatile methods for data cleaning, selection, and transformation. With pandas, researchers can easily load structured datasets, handle missing data, and conduct exploratory data analysis (EDA) to gain insights into the CKD data.

3. scikit-learn (sklearn): The Machine Learning Swiss Army Knife

scikit-learn, often referred to as sklearn, stands as a comprehensive **machine learning** library that encompasses an extensive range of tools for model development, evaluation, and deployment. In the context of CKD prediction, it serves as the primary workhorse for implementing machine learning algorithms, including logistic regression, decision trees, random forests, and support vector machines. sklearn offers modules for data preprocessing, model selection, and performance evaluation, streamlining the end-to-end process of CKD prediction.

4. Matplotlib and Seaborn: Visualizing Insights

Visualizations are a potent means of understanding CKD data and model performance. Matplotlib and Seaborn, popular Python libraries for data visualization, enable the creation of a wide array of charts, graphs, and plots. These visualizations aid in the exploration of CKD datasets, revealing patterns and relationships that might be hidden in raw data. For instance, data distributions, feature correlations, and model evaluation metrics can be effectively communicated through visually

appealing plots, enhancing both data understanding and model interpretation.

5. TensorFlow and Keras: Unleashing the Power of Deep Learning

In recent years, deep learning has made significant strides in healthcare applications, including CKD prediction. TensorFlow, an open-source deep learning framework developed by Google, and Keras, a high-level neural network API that runs on top of TensorFlow, are invaluable tools for researchers and data scientists seeking to harness the capabilities of deep neural networks. These libraries empower the development of complex neural architectures that can model intricate relationships within CKD data. Deep learning techniques are particularly well-suited for handling large, high-dimensional datasets, offering the potential for improved prediction accuracy.

6. XGBoost and LightGBM: Boosting Predictive Performance

Gradient boosting algorithms have gained prominence in the field of machine learning for their ability to deliver robust predictive models. In CKD prediction, libraries like XGBoost and LightGBM have become go-to choices for constructing ensemble models. These libraries leverage the power of gradient boosting, combining multiple decision trees to enhance model accuracy. XGBoost and LightGBM excel in handling tabular data, making them particularly suitable for CKD prediction tasks where structured clinical and demographic data are prevalent.

7. SHAP (SHapley Additive exPlanations): Illuminating Model Decisions

While predictive models hold immense potential for improving CKD diagnosis, their "black-box" nature can raise concerns about **transparency and interpretability.** SHAP, or SHapley Additive exPlanations, is a library that addresses this challenge by providing insights into how machine learning models arrive at their predictions. In CKD prediction projects, SHAP values can be employed to understand the importance of each feature in the model's decision-making process. This interpretability tool is a crucial component of ensuring that model predictions align with clinical knowledge and can be trusted by healthcare professionals.

8. Flask and Django: Building Interfaces for Deployment

The deployment of CKD prediction models into healthcare systems often involves creating user-friendly interfaces for healthcare providers. Flask and Django are web application frameworks that facilitate the development of such interfaces. Flask, known for its simplicity, is suitable for smaller projects, while Django offers a more comprehensive framework for larger applications. These frameworks enable the creation of web-based platforms where healthcare providers can input patient data and receive model predictions, enhancing the accessibility and usability of CKD prediction tools.

9. Docker: Containerization for Model Portability

Docker, a platform for containerizing applications, plays a pivotal role in deploying machine learning models, including those used for CKD prediction. Containerization allows models to be packaged

with all their dependencies, ensuring consistent behavior across different environments and healthcare systems. By encapsulating the model within a Docker container, researchers and data scientists can enhance the portability and reproducibility of their CKD prediction solutions.

10. Joblib and Pickle: Serialization for Model Persistence

Once a CKD prediction model is trained, it needs to be serialized for later use. Joblib and Pickle are Python libraries that facilitate the serialization and deserialization of machine learning models. These libraries allow trained models to be saved to disk and reloaded when needed, ensuring that the model can be easily integrated into healthcare systems and used for real-time predictions.

11. imbalanced-learn: Addressing Class Imbalance

In healthcare datasets, class imbalance is a common challenge, as some conditions, like CKD, may be relatively rare. The imbalanced-learn library provides tools for resampling data to address class imbalance. Techniques such as oversampling the minority class or undersampling the majority class can be applied to ensure that the predictive model maintains its ability to detect rare CKD cases while controlling for biases introduced by class imbalance.

12. Flask-RESTful and FastAPI: Creating APIs for Integration

To seamlessly integrate CKD prediction models into healthcare systems, researchers often build APIs (Application Programming Interfaces) that allow other software systems to interact with the model. Flask-RESTful and FastAPI are Python frameworks designed for developing RESTful APIs.

These APIs enable easy communication between the deployed model and healthcare systems, making it possible for patient data to be processed and predictions to be delivered in real-time.

These Python libraries collectively form a powerful toolkit for researchers and data scientists engaged in CKD prediction projects. Leveraging these tools, healthcare providers can enhance their diagnostic capabilities, deliver timely interventions, and ultimately improve patient outcomes. By combining the strengths of data-driven solutions and medical expertise, CKD prediction becomes not only achievable but also a substantial advancement in modern healthcare.

APPENDIX 2

jupyter Kidney_Disease_Prediction Last Checkpoint: 12/10/2023 (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Notebook saved Not Trusted Python 3 (ipykernel)

Run Code

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline

# For Filtering the warnings
import warnings
warnings.filterwarnings('ignore')
```

In [2]:
```
data = pd.read_csv('kidney_disease.csv')
```

In [3]:
```
data.head()
```

Out[3]:

	id	age	bp	sg	al	su	rbc	pc	pcc	ba	...	pcv	wc	rc	htn	dm	cad	appet	pe	ane	classification
0	0	48.0	80.0	1.020	1.0	0.0	NaN	normal	notpresent	notpresent	...	44	7800	5.2	yes	yes	no	good	no	no	ckd
1	1	7.0	50.0	1.020	4.0	0.0	NaN	normal	notpresent	notpresent	...	38	6000	NaN	no	no	no	good	no	no	ckd
2	2	62.0	80.0	1.010	2.0	3.0	normal	normal	notpresent	notpresent	...	31	7500	NaN	no	yes	no	poor	no	yes	ckd
3	3	48.0	70.0	1.005	4.0	0.0	normal	abnormal	present	notpresent	...	32	6700	3.9	yes	no	no	poor	yes	yes	ckd
4	4	51.0	80.0	1.010	2.0	0.0	normal	normal	notpresent	notpresent	...	35	7300	4.6	no	no	no	good	no	no	ckd

5 rows × 26 columns

ckd=chronic kidney disease

In [4]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 400 entries, 0 to 399
Data columns (total 26 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   id              400 non-null    int64
 1   age             391 non-null    float64
```

In [4]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 400 entries, 0 to 399
Data columns (total 26 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   id              400 non-null    int64
 1   age             391 non-null    float64
 2   bp              388 non-null    float64
 3   sg              353 non-null    float64
 4   al              354 non-null    float64
 5   su              351 non-null    float64
 6   rbc             248 non-null    object
 7   pc              335 non-null    object
 8   pcc             396 non-null    object
 9   ba              396 non-null    object
 10  bgr             356 non-null    float64
 11  bu              381 non-null    float64
 12  sc              383 non-null    float64
 13  sod             313 non-null    float64
 14  pot             312 non-null    float64
 15  hemo            348 non-null    float64
 16  pcv             330 non-null    object
 17  wc              295 non-null    object
 18  rc              270 non-null    object
 19  htn             398 non-null    object
 20  dm              398 non-null    object
 21  cad             398 non-null    object
 22  appet           399 non-null    object
 23  pe              399 non-null    object
 24  ane             399 non-null    object
 25  classification  400 non-null    object
dtypes: float64(11), int64(1), object(14)
memory usage: 81.4+ KB
```

In [26]:
```
data.classification=data.classification.replace("ckd\t","ckd")
```

In [27]:
```
data.classification.unique()
```

Out[27]:
```
array(['ckd', 'notckd'], dtype=object)
```

In [28]:
```
data.drop('id', axis = 1, inplace = True)
```

In [29]:
```
data.head()
```

Out[29]:

	age	bp	sg	al	su	rbc	pc	pcc	ba	bgr	...	pcv	wc	rc	htn	dm	cad	appet	pe	ane	classification
0	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
1	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
2	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
3	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
4	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

5 rows × 25 columns

In [30]:
```
data['classification'] = data['classification'].replace(['ckd','notckd'], [1,0])
```

In [31]:
```
data.head()
```

Out[31]:

	age	bp	sg	al	su	rbc	pc	pcc	ba	bgr	...	pcv	wc	rc	htn	dm	cad	appet	pe	ane	classification
0	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	1
1	7.0	50.0	1.020	4.0	0.0	NaN	normal	notpresent	notpresent	NaN	...	38	6000	NaN	no	no	no	good	no	no	1
2	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	1
3	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	1
4	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	1

5 rows × 25 columns

In [32]:
```
data.isnull().sum()
```

Out[32]:
```
age                 9
```

In [32]:
```
data.isnull().sum()
```

Out[32]:
```
age                 9
bp                 12
sg                 47
al                 46
su                 49
rbc               152
pc                 65
pcc                 4
ba                  4
bgr                44
bu                 19
sc                 17
sod                87
pot                88
hemo               52
pcv                70
wc                105
rc                130
htn                 2
dm                  2
cad                 2
appet               1
pe                  1
ane                 1
classification      0
dtype: int64
```

In [33]:
```
df = data.dropna(axis = 0)
print(f"Before dropping all NaN values: {data.shape}")
print(f"After dropping all NaN values: {df.shape}")
```

```
Before dropping all NaN values: (400, 25)
After dropping all NaN values: (158, 25)
```

In [34]:
```
df.head()
```

Out[34]:

	age	bp	sg	al	su	rbc	pc	pcc	ba	bgr	...	pcv	wc	rc	htn	dm	cad	appet	pe	ane	classification
3	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	1

jupyter Kidney_Disease_Prediction Last Checkpoint: 12/10/2023 (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3 (ipykernel)

Run Code

In [34]:
```
df.head()
```

Out[34]:

	age	bp	sg	al	su	rbc	pc	pcc	ba	bgr	...	pcv	wc	rc	htn	dm	cad	appet	pe	ane	classification
3	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	1
9	53.0	90.0	1.020	2.0	0.0	abnormal	abnormal	present	notpresent	70.0	...	29	12100	3.7	yes	yes	no	poor	no	yes	1
11	63.0	70.0	1.010	3.0	0.0	abnormal	abnormal	present	notpresent	380.0	...	32	4500	3.8	yes	yes	no	poor	yes	no	1
14	68.0	80.0	1.010	3.0	2.0	normal	abnormal	present	present	157.0	...	16	11000	2.6	yes	yes	yes	poor	yes	no	1
20	61.0	80.0	1.015	2.0	0.0	abnormal	abnormal	notpresent	notpresent	173.0	...	24	9200	3.2	yes	yes	yes	poor	yes	yes	1

5 rows × 25 columns

In [35]:
```
df.index = range(0,len(df),1)
df.head()
```

Out[35]:

	age	bp	sg	al	su	rbc	pc	pcc	ba	bgr	...	pcv	wc	rc	htn	dm	cad	appet	pe	ane	classification
0	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	1
1	53.0	90.0	1.020	2.0	0.0	abnormal	abnormal	present	notpresent	70.0	...	29	12100	3.7	yes	yes	no	poor	no	yes	1
2	63.0	70.0	1.010	3.0	0.0	abnormal	abnormal	present	notpresent	380.0	...	32	4500	3.8	yes	yes	no	poor	yes	no	1
3	68.0	80.0	1.010	3.0	2.0	normal	abnormal	present	present	157.0	...	16	11000	2.6	yes	yes	yes	poor	yes	no	1
4	61.0	80.0	1.015	2.0	0.0	abnormal	abnormal	notpresent	notpresent	173.0	...	24	9200	3.2	yes	yes	yes	poor	yes	yes	1

5 rows × 25 columns

In [36]:
```
for i in df['wc']:
    print(i)
```

```
6700
12100
4500
11000
9200
6900
9600
18900
```

In [37]:
```
df['wc']=df['wc'].replace(["\t6200","\t8400"],[6200,8400])
```

In [38]:
```
for i in df['wc']:
    print(i)
```

```
7000
15200
5000
16300
8400
10500
15200
14600
7900
10900
12800
12400
19100
7500
16700
9600
26400
9800
8800
7400
```

In [39]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 158 entries, 0 to 157
Data columns (total 25 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   age             158 non-null    float64
 1   bp              158 non-null    float64
 2   sg              158 non-null    float64
 3   al              158 non-null    float64
 4   su              158 non-null    float64
 5   rbc             158 non-null    object
 6   pc              158 non-null    object
 7   pcc             158 non-null    object
```

In [40]:
```
df['pcv']=df['pcv'].astype(int)
df['wc']=df['wc'].astype(int)
df['rc']=df['rc'].astype(float)
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 158 entries, 0 to 157
Data columns (total 25 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   age             158 non-null    float64
 1   bp              158 non-null    float64
 2   sg              158 non-null    float64
 3   al              158 non-null    float64
 4   su              158 non-null    float64
 5   rbc             158 non-null    object 
 6   pc              158 non-null    object 
 7   pcc             158 non-null    object 
 8   ba              158 non-null    object 
 9   bgr             158 non-null    float64
 10  bu              158 non-null    float64
 11  sc              158 non-null    float64
 12  sod             158 non-null    float64
 13  pot             158 non-null    float64
 14  hemo            158 non-null    float64
 15  pcv             158 non-null    int32  
 16  wc              158 non-null    int32  
 17  rc              158 non-null    float64
 18  htn             158 non-null    object 
 19  dm              158 non-null    object 
 20  cad             158 non-null    object 
 21  appet           158 non-null    object 
 22  pe              158 non-null    object 
 23  ane             158 non-null    object 
 24  classification  158 non-null    int64  
dtypes: float64(12), int32(2), int64(1), object(10)
memory usage: 29.8+ KB
```

In [41]:
```
object_dtypes = df.select_dtypes(include = 'object')
object_dtypes.head()
```

Out[41]:

	rbc	pc	pcc	ba	htn	dm	cad	appet	pe	ane
0	normal	abnormal	present	notpresent	yes	no	no	poor	yes	yes
1	abnormal	abnormal	present	notpresent	yes	yes	no	poor	no	yes
2	abnormal	abnormal	present	notpresent	yes	yes	no	poor	yes	no
3	normal	abnormal	present	present	yes	yes	yes	poor	yes	no
4	abnormal	abnormal	notpresent	notpresent	yes	yes	yes	poor	yes	yes

In [42]:
```
dictonary = {
        "rbc": {
        "abnormal":1,
        "normal": 0,
    },
        "pc":{
        "abnormal":1,
        "normal": 0,
    },
        "pcc":{
        "present":1,
        "notpresent":0,
    },
        "ba":{
        "notpresent":0,
        "present": 1,
    },
        "htn":{
        "yes":1,
        "no": 0,
    },
        "dm":{
        "yes":1,
        "no":0,
    },
        "cad":{
```

```
    },
        "cad":{
        "yes":1,
        "no": 0,
    },
        "appet":{
        "good":1,
        "poor": 0,
    },
        "pe":{
        "yes":1,
        "no":0,
    },
        "ane":{
        "yes":1,
        "no":0,
    }
}
```

In [43]:
```
df=df.replace(dictonary)
```

In [44]:
```
df.head()
```

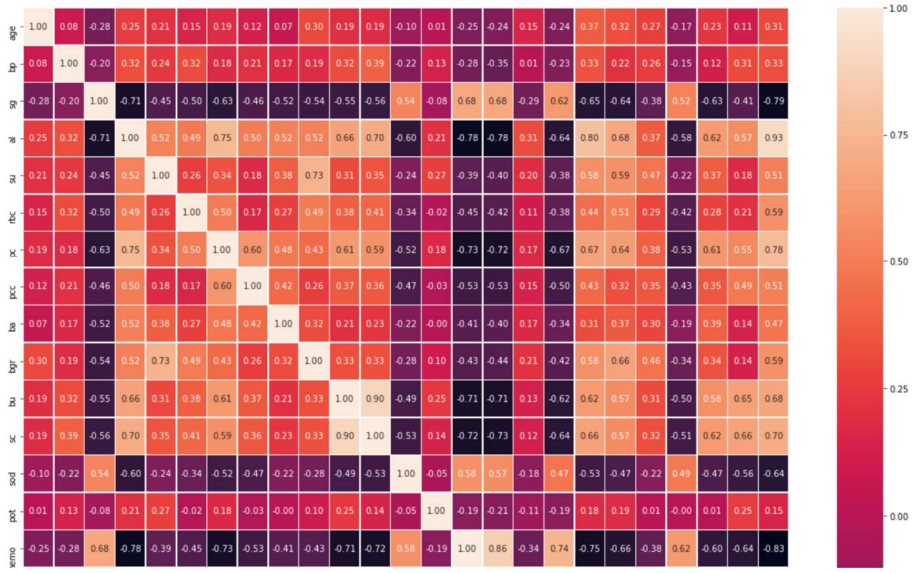
Out[44]:

	age	bp	sg	al	su	rbc	pc	pcc	ba	bgr	...	pcv	wc	rc	htn	dm	cad	appet	pe	ane	classification
0	48.0	70.0	1.005	4.0	0.0	0	1	1	0	117.0	...	32	6700	3.9	1	0	0	0	1	1	1
1	53.0	90.0	1.020	2.0	0.0	1	1	1	0	70.0	...	29	12100	3.7	1	1	0	0	0	1	1
2	63.0	70.0	1.010	3.0	0.0	1	1	1	0	380.0	...	32	4500	3.8	1	1	0	0	1	0	1
3	68.0	80.0	1.010	3.0	2.0	0	1	1	1	157.0	...	16	11000	2.6	1	1	1	0	1	0	1
4	61.0	80.0	1.015	2.0	0.0	1	1	0	0	173.0	...	24	9200	3.2	1	1	1	0	1	1	1

5 rows × 25 columns

In [45]:
```
import seaborn as sns
plt.figure(figsize = (20,20))
sns.heatmap(df.corr(), annot = True, fmt=".2f",linewidths=0.5)
```

Out[45]: <AxesSubplot:>



In [46]:
```
df.corr()
```

Out[46]:

	age	bp	sg	al	su	rbc	pc	pcc	ba	bgr	...	pcv	wc	rc
age	1.000000	0.079712	-0.277303	0.253380	0.207711	0.147971	0.188907	0.124032	0.068353	0.301915	...	-0.235116	0.153132	-0.242235
bp	0.079712	1.000000	-0.198897	0.322507	0.243828	0.316670	0.179834	0.206507	0.174555	0.190113	...	-0.353504	0.008274	-0.228533
sg	-0.277303	-0.198897	1.000000	-0.712331	-0.448477	-0.500494	-0.630323	-0.460050	-0.516392	-0.544781	...	0.678472	-0.288930	0.619092
al	0.253380	0.322507	-0.712331	1.000000	0.521448	0.489941	0.752956	0.503341	0.516104	0.518123	...	-0.775528	0.314574	-0.640099
su	0.207711	0.243828	-0.448477	0.521448	1.000000	0.256568	0.335901	0.177327	0.381929	0.730050	...	-0.404821	0.201000	-0.377726
rbc	0.147971	0.316670	-0.500494	0.489941	0.256568	1.000000	0.498959	0.168592	0.273177	0.493857	...	-0.422537	0.108857	-0.379378
pc	0.188907	0.179834	-0.630323	0.752956	0.335901	0.498959	1.000000	0.600092	0.481227	0.430646	...	-0.718042	0.169936	-0.667113
pcc	0.124032	0.206507	-0.460050	0.503341	0.177327	0.168592	0.600092	1.000000	0.415033	0.257768	...	-0.534564	0.146742	-0.499401
ba	0.068353	0.174555	-0.516392	0.516104	0.381929	0.273177	0.481227	0.415033	1.000000	0.318095	...	-0.397500	0.170071	-0.343299
bgr	0.301915	0.190113	-0.544781	0.518123	0.730050	0.493857	0.430646	0.257768	0.318095	1.000000	...	-0.443818	0.212093	-0.418085
bu	0.190636	0.316287	-0.545319	0.661940	0.312259	0.378478	0.613318	0.366726	0.205351	0.326496	...	-0.706582	0.128961	-0.621456
sc	0.189721	0.386551	-0.563122	0.702889	0.347196	0.410408	0.588517	0.361965	0.229238	0.331284	...	-0.726187	0.123953	-0.639021
sod	-0.102933	-0.224710	0.539285	-0.599334	-0.242491	-0.344916	-0.520324	-0.473954	-0.221374	-0.284968	...	0.570045	-0.176238	0.465125
pot	0.006866	0.127801	-0.075057	0.209492	0.271954	-0.019319	0.176150	-0.030297	-0.000279	0.102226	...	-0.213488	-0.107559	-0.193783
hemo	-0.245645	-0.282365	0.682086	-0.784745	-0.385511	-0.452566	-0.733140	-0.531182	-0.410353	-0.434158	...	0.856775	-0.337435	0.743999
pcv	-0.235116	-0.353504	0.678472	-0.775528	-0.404821	-0.422537	-0.718042	-0.534564	-0.397500	-0.443818	...	1.000000	-0.349607	0.739019
wc	0.153132	0.008274	-0.288930	0.314574	0.201000	0.108857	0.169936	0.146742	0.170071	0.212093	...	-0.349607	1.000000	-0.272390
rc	-0.242235	-0.228533	0.619092	-0.640099	-0.377726	-0.379378	-0.667113	-0.499401	-0.343299	-0.418085	...	0.739019	-0.272390	1.000000
htn	0.372348	0.334951	-0.648168	0.796876	0.577286	0.442400	0.666767	0.432876	0.314961	0.579407	...	-0.752043	0.223916	-0.671740
dm	0.323957	0.218096	-0.639391	0.678582	0.591010	0.511777	0.636288	0.321900	0.367477	0.663012	...	-0.655039	0.287010	-0.594881
cad	0.269868	0.257709	-0.379305	0.374755	0.466658	0.293269	0.384223	0.352255	0.297063	0.459164	...	-0.375627	0.021259	-0.362439
appet	-0.170259	-0.145047	0.523944	-0.578080	-0.220547	-0.418639	-0.528435	-0.432515	-0.187815	-0.338924	...	0.629102	-0.328730	0.556182
pe	0.232327	0.117878	-0.633622	0.622268	0.374128	0.282868	0.606234	0.350171	0.393819	0.336141	...	-0.606829	0.282628	-0.566384

In [47]:
```
X = df.drop(['classification', 'sg', 'appet', 'rc', 'pcv', 'hemo', 'sod'], axis = 1)
y = df['classification']
```

In [48]:
```
X.columns
```

Out[48]:
```
Index(['age', 'bp', 'al', 'su', 'rbc', 'pc', 'pcc', 'ba', 'bgr', 'bu', 'sc',
       'pot', 'wc', 'htn', 'dm', 'cad', 'pe', 'ane'],
      dtype='object')
```

In [49]:
```
from sklearn.model_selection import train_test_split
```

In [50]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 42)
```

In [51]:
```
from sklearn.ensemble import RandomForestClassifier
```

In [52]:
```
model = RandomForestClassifier(n_estimators = 20)
model.fit(X_train, y_train)
```

Out[52]:
```
RandomForestClassifier(n_estimators=20)
```

In [53]:
```
from sklearn.metrics import confusion_matrix, accuracy_score
```

In [54]:
```
confusion_matrix(y_test, model.predict(X_test))
```

Out[54]:
```
array([[23,  0],
       [ 0,  9]], dtype=int64)
```

In [55]:
```
print(f"Accuracy is {round(accuracy_score(y_test, model.predict(X_test))*100, 2)}%")
```

```
Accuracy is 100.0%
```

In [56]:
```
import pickle
pickle.dump(model, open('kidney.pkl', 'wb'))
```

jupyter Main-Code Last Checkpoint: 12/10/2023 (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

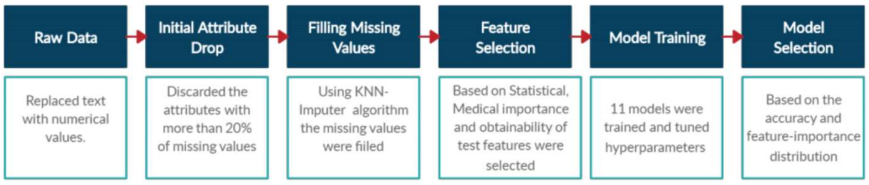
Not Trusted | Python 3 (ipykernel)

Run Markdown

Machine Learning Approach to Predict the Chronic Kidney Disease

It identifies the limitations in handling missing values when analysing CKD data, proposes a new method to handle missing values and presents the evaluation of different methods based on UCI dataset. Further, this work also highlights the importance of statistical analysis as well as the domain knowledge of the features when making a prediction based on clinical data related to CKD.

chronic renal disease, machine learning, classification algorithms, extra tree classifier, random forest classifier,XGBoost



Data set we contains

1. age - age
2. bp - blood pressure
3. sg - specific gravity
4. al - albumin
5. su - sugar
6. rbc - red blood cells
7. pc - pus cell
8. pcc - pus cell clumps
9. ba - bacteria
10. bgr - blood glucose random
11. bu - blood urea
12. sc - serum creatinine
13. sod - sodium
14. pot - potassium
15. hemo - haemoglobin
16. pcv - packed cell volume
17. wc - white blood cell count
18. rc - red blood cell count
19. htn - hypertension
20. dm - diabetes mellitus
21. cad - coronary artery disease
22. appet - appetite
23. pe - pedal edema
24. ane - anemia
25. lassification - class

Feature description of the Data

1. Age(numerical) --> age in years
2. Blood Pressure(numerical) bp in mm/Hg
3. Specific Gravity(nominal) sg - (1.005,1.010,1.015,1.020,1.025)
4. Albumin(nominal)al - (0,1,2,3,4,5)
5. Sugar(nominal) su - (0,1,2,3,4,5)
6. Red Blood Cells(nominal) rbc - (normal,abnormal)
7. Pus Cell (nominal)pc - (normal,abnormal)
8. Pus Cell clumps(nominal)pcc - (present,notpresent)
9. Bacteria(nominal) ba - (present,notpresent)
10. Blood Glucose Random(numerical) bgr in mgs/dl
11. Blood Urea(numerical) bu in mgs/dl
12. Serum Creatinine(numerical) sc in mgs/dl
13. Sodium(numerical) sod in mEq/L
14. Potassium(numerical) pot in mEq/L
15. Haemoglobin(numerical) hemo in gms
16. Packed Cell Volume(numerical)
17. White Blood Cell Count(numerical) wc in cells/cumm
18. Red Blood Cell Count(numerical) rc in millions/cmm
19. Hypertension(nominal) htn - (yes,no)
20. Diabetes Mellitus(nominal) dm - (yes,no)
21. Coronary Artery Disease(nominal) cad - (yes,no)
22. Appetite(nominal) ppet - (good,poor)
23. Pedal Edema(nominal) pe - (yes,no)
24. Anemia(nominal)ane - (yes,no)
25. Class (nominal) class - (ckd,notckd)

Install & Importing the Dependencies

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns

import warnings
warnings.filterwarnings('ignore')
```

Extract the Data-Set (Kidney_Disease.csv)

In [2]:
```
kidney=pd.read_csv('kidney_disease.csv')
```

In [3]:
```
kidney.shape
```

Out[3]:
```
(400, 26)
```

- It Indicates there are about **400** Rows and **26** Columns are present in our Data Set

In [4]:
```
kidney.head()
```

Out[4]:

	id	age	bp	sg	al	su	rbc	pc	pcc	ba	...	pcv	wc	rc	htn	dm	cad	appet	pe	ane	classification
0	0	48.0	80.0	1.020	1.0	0.0	NaN	normal	notpresent	notpresent	...	44	7800	5.2	yes	yes	no	good	no	no	ckd
1	1	7.0	50.0	1.020	4.0	0.0	NaN	normal	notpresent	notpresent	...	38	6000	NaN	no	no	no	good	no	no	ckd
2	2	62.0	80.0	1.010	2.0	3.0	normal	normal	notpresent	notpresent	...	31	7500	NaN	no	yes	no	poor	no	yes	ckd
3	3	48.0	70.0	1.005	4.0	0.0	normal	abnormal	present	notpresent	...	32	6700	3.9	yes	no	no	poor	yes	yes	ckd
4	4	51.0	80.0	1.010	2.0	0.0	normal	normal	notpresent	notpresent	...	35	7300	4.6	no	no	no	good	no	no	ckd

5 rows × 26 columns

In [5]:
```
kidney.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 400 entries, 0 to 399
Data columns (total 26 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   id              400 non-null    int64  
 1   age             391 non-null    float64
 2   bp              388 non-null    float64
 3   sg              353 non-null    float64
 4   al              354 non-null    float64
 5   su              351 non-null    float64
 6   rbc             248 non-null    object 
 7   pc              335 non-null    object 
 8   pcc             396 non-null    object 
 9   ba              396 non-null    object 
 10  bgr             356 non-null    float64
 11  bu              381 non-null    float64
 12  sc              383 non-null    float64
 13  sod             313 non-null    float64
 14  pot             312 non-null    float64
 15  hemo            348 non-null    float64
 16  pcv             330 non-null    object 
 17  wc              295 non-null    object 
 18  rc              270 non-null    object 
 19  htn             398 non-null    object 
 20  dm              398 non-null    object 
 21  cad             398 non-null    object 
 22  appet           399 non-null    object 
 23  pe              399 non-null    object 
 24  ane             399 non-null    object 
 25  classification  400 non-null    object 
dtypes: float64(11), int64(1), object(14)
memory usage: 81.4+ KB
```

In [6]:
```
kidney.describe()
```

Out[6]:

	id	age	bp	sg	al	su	bgr	bu	sc	sod	pot	hemo
count	400.000000	391.000000	388.000000	353.000000	354.000000	351.000000	356.000000	381.000000	383.000000	313.000000	312.000000	348.000000
mean	199.500000	51.483376	76.469072	1.017408	1.016949	0.450142	148.036517	57.425722	3.072454	137.528754	4.627244	12.526437
std	115.614301	17.169714	13.683637	0.005717	1.352679	1.099191	79.281714	50.503006	5.741126	10.408752	3.193904	2.912587
min	0.000000	2.000000	50.000000	1.005000	0.000000	0.000000	22.000000	1.500000	0.400000	4.500000	2.500000	3.100000
25%	99.750000	42.000000	70.000000	1.010000	0.000000	0.000000	99.000000	27.000000	0.900000	135.000000	3.800000	10.300000
50%	199.500000	55.000000	80.000000	1.020000	0.000000	0.000000	121.000000	42.000000	1.300000	138.000000	4.400000	12.650000
75%	299.250000	64.500000	80.000000	1.020000	2.000000	0.000000	163.000000	66.000000	2.800000	142.000000	4.900000	15.000000
max	399.000000	90.000000	180.000000	1.025000	5.000000	5.000000	490.000000	391.000000	76.000000	163.000000	47.000000	17.800000

Performing Exploitory Data Analysis (EDA)

- Modifying the Column Names as per our requirements

In [7]:
```
columns=pd.read_csv("data_description.txt",sep='-')
columns=columns.reset_index()
```

In [8]:
```
columns.columns=['cols','abb_col_names']
```

In [9]:
```
columns
```

Out[9]:

	cols	abb_col_names
0	id	id
1	age	age
2	bp	blood pressure
3	sg	specific gravity
4	al	albumin
5	su	sugar

6	rbc	red blood cells
7	pc	pus cell
8	pcc	pus cell clumps
9	ba	bacteria
10	bgr	blood glucose random
11	bu	blood urea
12	sc	serum creatinine
13	sod	sodium
14	pot	potassium
15	hemo	haemoglobin
16	pcv	packed cell volume
17	wc	white blood cell count
18	rc	red blood cell count
19	htn	ypertension
20	dm	diabetes mellitus
21	cad	coronary artery disease
22	appet	appetite
23	pe	pedal edema
24	ane	anemia
25	classification	class

In [10]:
```
kidney.head()
```

Out[10]:

	id	age	bp	sg	al	su	rbc	pc	pcc	ba	...	pcv	wc	rc	htn	dm	cad	appet	pe	ane	classification
0	0	48.0	80.0	1.020	1.0	0.0	NaN	normal	notpresent	notpresent	...	44	7800	5.2	yes	yes	no	good	no	no	ckd
1	1	7.0	50.0	1.020	4.0	0.0	NaN	normal	notpresent	notpresent	...	38	6000	NaN	no	no	no	good	no	no	ckd
2	2	62.0	80.0	1.010	2.0	3.0	normal	normal	notpresent	notpresent	...	31	7500	NaN	no	yes	no	poor	no	yes	ckd
3	3	48.0	70.0	1.005	4.0	0.0	normal	abnormal	present	notpresent	...	32	6700	3.9	yes	no	no	poor	yes	yes	ckd
4	4	51.0	80.0	1.010	2.0	0.0	normal	normal	notpresent	notpresent	...	35	7300	4.6	no	no	no	good	no	no	ckd

5 rows × 26 columns

In [11]:
```
kidney.columns=columns['abb_col_names'].values
```

In [12]:
```
kidney.head()
```

Out[12]:

	id	age	blood pressure	specific gravity	albumin	sugar	red blood cells	pus cell	pus cell clumps	bacteria	...	packed cell volume	white blood cell count	red blood cell count	ypertension	diabetes mellitus	coronary artery disease	appetite
0	0	48.0	80.0	1.020	1.0	0.0	NaN	normal	notpresent	notpresent	...	44	7800	5.2	yes	yes	no	good
1	1	7.0	50.0	1.020	4.0	0.0	NaN	normal	notpresent	notpresent	...	38	6000	NaN	no	no	no	good
2	2	62.0	80.0	1.010	2.0	3.0	normal	normal	notpresent	notpresent	...	31	7500	NaN	no	yes	no	poor
3	3	48.0	70.0	1.005	4.0	0.0	normal	abnormal	present	notpresent	...	32	6700	3.9	yes	no	no	poor
4	4	51.0	80.0	1.010	2.0	0.0	normal	normal	notpresent	notpresent	...	35	7300	4.6	no	no	no	good

5 rows × 26 columns

In [13]:
```
kidney.describe().T
```

Out[13]:

	count	mean	std	min	25%	50%	75%	max
id	400.0	199.500000	115.614301	0.000	99.75	199.50	299.25	399.000
age	391.0	51.483376	17.169714	2.000	42.00	55.00	64.50	90.000

blood pressure	388.0	76.469072	13.683637	50.000	70.00	80.00	80.00	180.000
specific gravity	353.0	1.017408	0.005717	1.005	1.01	1.02	1.02	1.025
albumin	354.0	1.016949	1.352679	0.000	0.00	0.00	2.00	5.000
sugar	351.0	0.450142	1.099191	0.000	0.00	0.00	0.00	5.000
blood glucose random	356.0	148.036517	79.281714	22.000	99.00	121.00	163.00	490.000
blood urea	381.0	57.425722	50.503006	1.500	27.00	42.00	66.00	391.000
serum creatinine	383.0	3.072454	5.741126	0.400	0.90	1.30	2.80	76.000
sodium	313.0	137.528754	10.408752	4.500	135.00	138.00	142.00	163.000
potassium	312.0	4.627244	3.193904	2.500	3.80	4.40	4.90	47.000
haemoglobin	348.0	12.526437	2.912587	3.100	10.30	12.65	15.00	17.800

In [14]:
```
def convert_dtype(kidney,feature):
    kidney[feature]=pd.to_numeric(kidney[feature],errors='coerce')    #whereever we have Nan values , this errors pa
```

In [15]:
```
features=['packed cell volume','white blood cell count','red blood cell count']
for i in features:
    convert_dtype(kidney,i)
```

In [16]:
```
kidney.dtypes
```

Out[16]:
```
id                              int64
age                           float64
blood pressure                float64
specific gravity              float64
albumin                       float64
sugar                         float64
red blood cells                object
 pus cell                      object
pus cell clumps                object
bacteria                       object
blood glucose random          float64
blood urea                    float64
serum creatinine              float64
sodium                        float64
haemoglobin                   float64
packed cell volume            float64
white blood cell count        float64
red blood cell count          float64
ypertension                    object
diabetes mellitus              object
coronary artery disease        object
appetite                       object
pedal edema                    object
anemia                         object
class                          object
dtype: object
```

In [17]:
```
kidney.drop('id',inplace=True,axis=1)
```

Performing Data cleaning

In [18]:
```
def extract_cat_num(kidney):
    cat_col=[col for col in kidney.columns if kidney[col].dtype=='O']
    num_col=[col for col in kidney.columns if kidney[col].dtype!='O']
    return cat_col,num_col
```

In [19]:
```
cat_col,num_col=extract_cat_num(kidney)
```

In [20]:
```
cat_col
```

Out[20]:
```
['red blood cells',
 ' pus cell',
 'pus cell clumps',
 'bacteria',
 'ypertension',
 'diabetes mellitus',
 'coronary artery disease',
 'appetite',
 'pedal edema',
 'anemia',
 'class']
```

In [21]:
```
num_col
```

Out[21]:
```
['age',
 'blood pressure',
 'specific gravity',
 'albumin',
 'sugar',
 'blood glucose random',
 'blood urea',
 'serum creatinine',
 'sodium',
 'potassium',
 'haemoglobin',
 'packed cell volume',
 'white blood cell count',
 'red blood cell count']
```

In [22]:
```
# dirtiness in categorical data
for col in cat_col:
    print('{} has {} values'.format(col,kidney[col].unique()))
    print("\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


ypertension 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


pedal edema has ['no' 'yes' nan] values


anemia has ['no' 'yes' nan] values


class has ['ckd' 'ckd\t' 'notckd'] values
```

In [23]:
```
kidney['diabetes mellitus'].replace(to_replace={'\tno':'no','\tyes':'yes'},inplace=True)
kidney['coronary artery disease'].replace(to_replace={'\tno':'no'},inplace=True)
kidney['class'].replace(to_replace={'ckd\t':'ckd'},inplace=True)
```

In [24]:
```
# no dirtiness
for col in cat_col:
    print('{} has {} values'.format(col,kidney[col].unique()))
    print("\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


ypertension has ['yes' 'no' nan] values
```

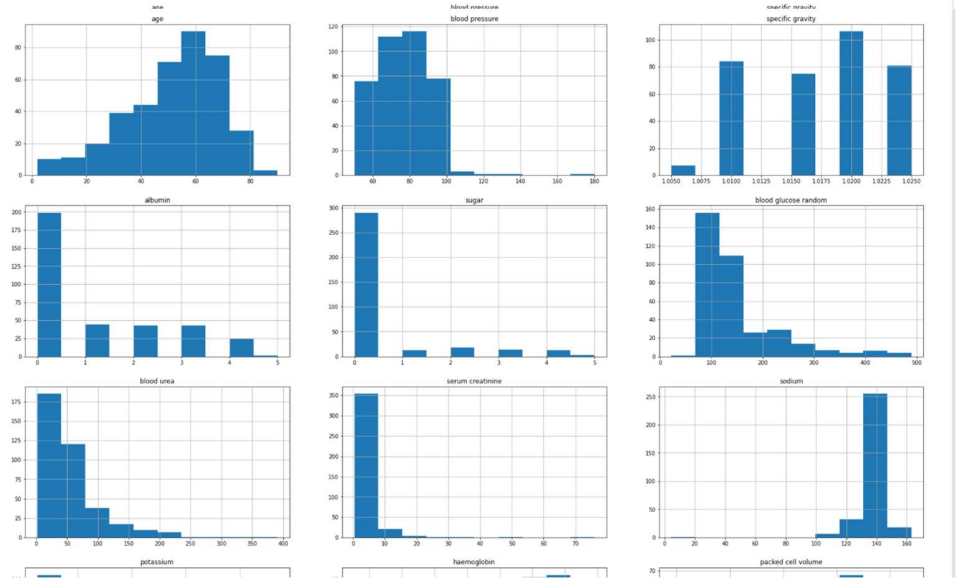
```
diabetes mellitus has ['yes' 'no' ' yes' nan] values

coronary artery disease has ['no' 'yes' nan] values

appetite has ['good' 'poor' nan] values

pedal edema has ['no' 'yes' nan] values

anemia has ['no' 'yes' nan] values

class has ['ckd' 'notckd'] values
```

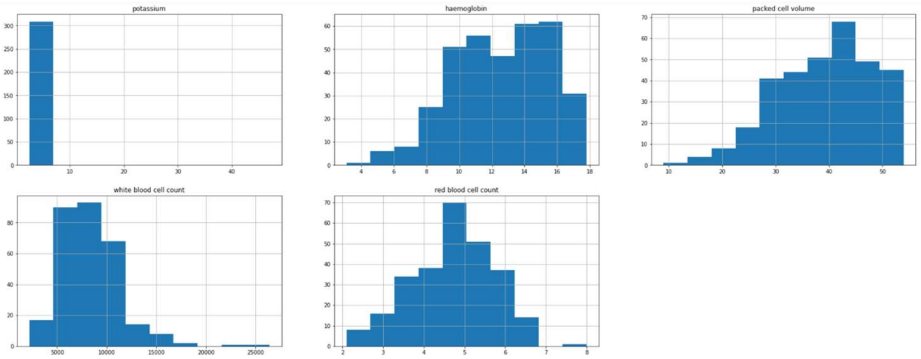
Analysing distribution of each and every column

In [25]:
```
len(num_col)
```

Out[25]: 14

In [26]:
```
plt.figure(figsize=(30,30))
for i,feature in enumerate(num_col):
    plt.subplot(5,3,i+1)   # 5 rows and 3 columns
    kidney[feature].hist()
    plt.title(feature)
```



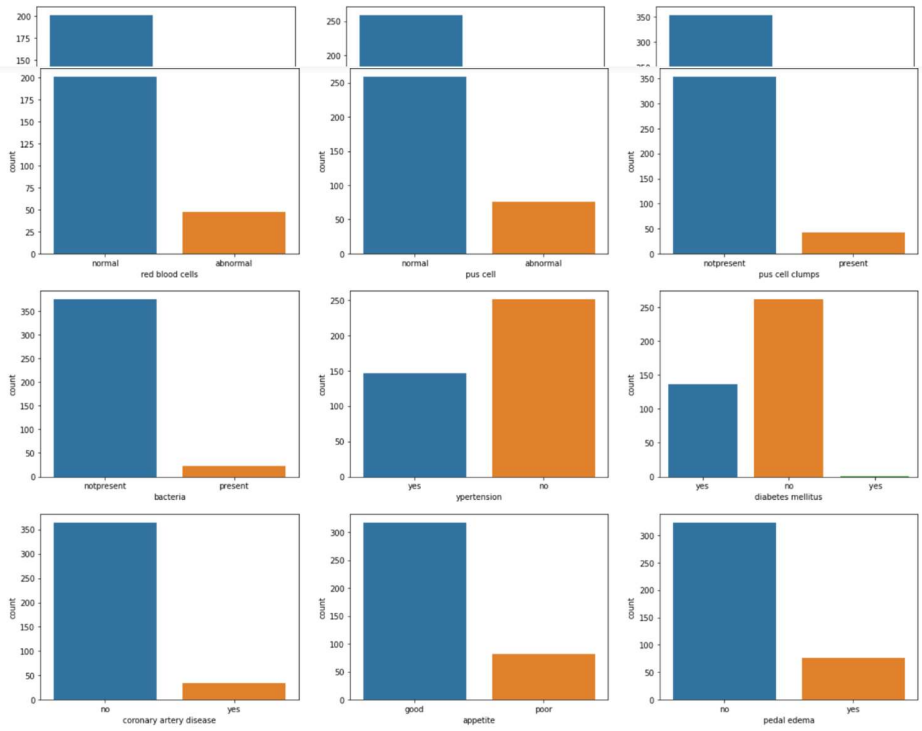


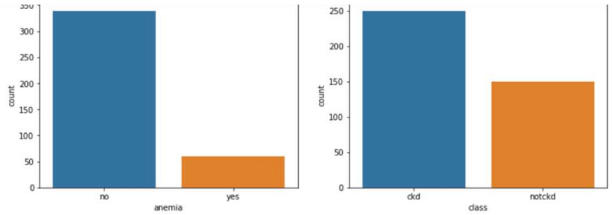
Check Label distribution of categorical Data

In [27]:
```
len(cat_col)
```

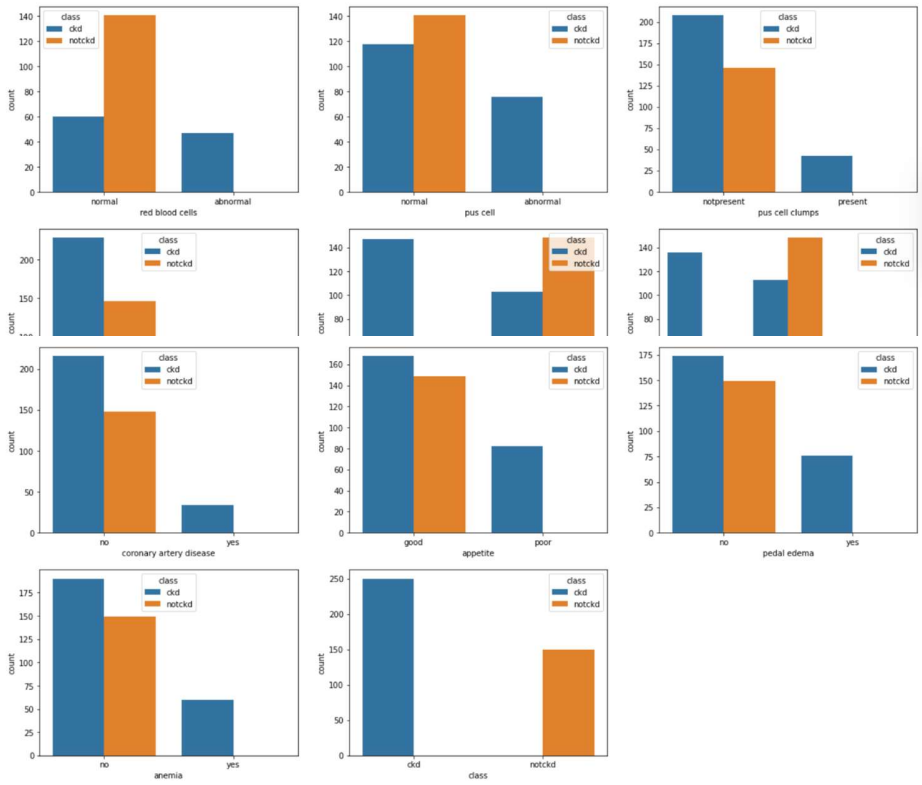
Out[27]: 11

In [28]:
```
plt.figure(figsize=(20,20))

for i,feature in enumerate(cat_col):
    plt.subplot(4,3,i+1)
    sns.countplot(kidney[feature])
```



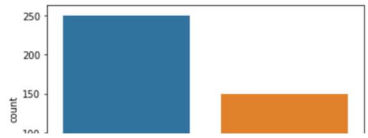


In [29]:
```
plt.figure(figsize=(20,20))

for i,feature in enumerate(cat_col):
    plt.subplot(4,3,i+1)
    sns.countplot(kidney[feature],hue=kidney['class'])
```



In [30]:
```
sns.countplot(kidney['class'])
```

Out[30]: <AxesSubplot:xlabel='class', ylabel='count'>



Correlation between features

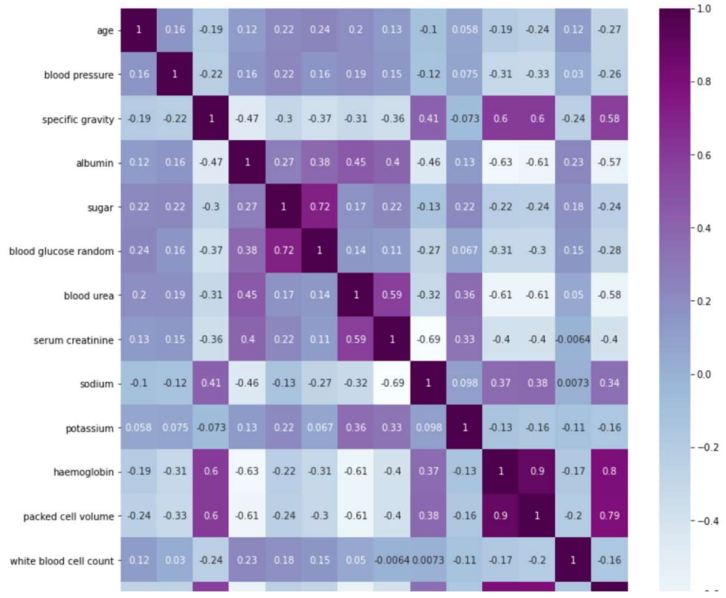
In [31]:
```
kidney.corr()
```

Out[31]:

	age	blood pressure	specific gravity	albumin	sugar	blood glucose random	blood urea	serum creatinine	sodium	potassium	haemoglobin	packed cell volume	white blood cell count
age	1.000000	0.159480	-0.191096	0.122091	0.220866	0.244992	0.196985	0.132531	-0.100046	0.058377	-0.192928	-0.242119	0.118339
blood pressure	0.159480	1.000000	-0.218836	0.160689	0.222576	0.160193	0.188517	0.146222	-0.116422	0.075151	-0.306540	-0.326319	0.029753
specific gravity	-0.191096	-0.218836	1.000000	-0.469760	-0.296234	-0.374710	-0.314295	-0.361473	0.412190	-0.072787	0.602582	0.603560	-0.236215
albumin	0.122091	0.160689	-0.469760	1.000000	0.269305	0.379464	0.453528	0.399198	-0.459896	0.129038	-0.634632	-0.611891	0.231989
sugar	0.220866	0.222576	-0.296234	0.269305	1.000000	0.717827	0.168583	0.223244	-0.131776	0.219450	-0.224775	-0.239189	0.184893
blood glucose random	0.244992	0.160193	-0.374710	0.379464	0.717827	1.000000	0.143322	0.114875	-0.267848	0.066966	-0.306189	-0.301385	0.150015
blood urea	0.196985	0.188517	-0.314295	0.453528	0.168583	0.143322	1.000000	0.586368	-0.323054	0.357049	-0.610360	-0.607621	0.050462
serum creatinine	0.132531	0.146222	-0.361473	0.399198	0.223244	0.114875	0.586368	1.000000	-0.690158	0.326107	-0.401670	-0.404193	-0.006390
sodium	-0.100046	-0.116422	0.412190	-0.459896	-0.131776	-0.267848	-0.323054	-0.690158	1.000000	0.097887	0.365183	0.376914	0.007277
potassium	0.058377	0.075151	-0.072787	0.129038	0.219450	0.066966	0.357049	0.326107	0.097887	1.000000	-0.133746	-0.163182	-0.105576
haemoglobin	-0.192928	-0.306540	0.602582	-0.634632	-0.224775	-0.306189	-0.610360	-0.401670	0.365183	-0.133746	1.000000	0.895382	-0.169413
packed cell volume	-0.242119	-0.326319	0.603560	-0.611891	-0.239189	-0.301385	-0.607621	-0.404193	0.376914	-0.163182	0.895382	1.000000	-0.197022
white blood cell count	0.118339	0.029753	-0.236215	0.231989	0.184893	0.150015	0.050462	-0.006390	0.007277	-0.105576	-0.169413	-0.197022	1.000000
red blood cell count	-0.268896	-0.261936	0.579476	-0.566437	-0.237448	-0.281541	-0.579087	-0.400852	0.344873	-0.158309	0.798880	0.791625	-0.158163

In [32]:
```
plt.figure(figsize=(12,12))
sns.heatmap(kidney.corr(method='pearson'),cbar=True,cmap='BuPu',annot=True)
```

Out[32]: <AxesSubplot:>



- Rbc count is positively correlated with specific gravity,haemoglobin,packed cell volume
- Rbc count is negatively correlated with albumin, blood urea
- Packed cell volume and haemoglobin are highly positive correlated
- Packed cell volume is negatively correlated with albumin and blood urea
- haemoglobin and albumin are negatively correlated

In [33]:
```
kidney.groupby(['red blood cells','class'])['red blood cell count'].agg(['count','mean','median','min','max'])
```

Out[33]:

red blood cells	class	count	mean	median	min	max
abnormal	ckd	25	3.832000	3.7	2.5	5.6
normal	ckd	40	3.782500	3.8	2.1	8.0
	notckd	134	5.368657	5.3	4.4	6.5

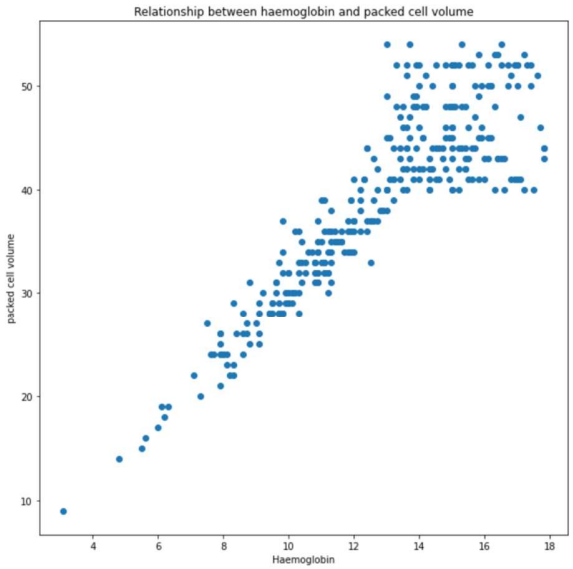
We can observe that when a person is not diseased its rbc count is 134, mean is also high whereas when he is diseased count drop downs to 25-40 and mean is low.

Relationship between haemoglobin and packed cell volume

In [34]:
```
plt.figure(figsize=(10,10))
plt.scatter(x=kidney.haemoglobin,y=kidney['packed cell volume'])
plt.xlabel('Haemoglobin')
plt.ylabel('packed cell volume')
plt.title('Relationship between haemoglobin and packed cell volume')
```

Out[34]:
```
Text(0.5, 1.0, 'Relationship between haemoglobin and packed cell volume')
```

Out[34]: Text(0.5, 1.0, 'Relationship between haemoglobin and packed cell volume')

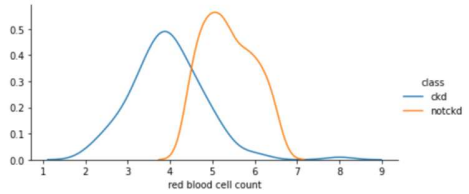


We can see that there is a linear relationship between haemoglobin and pacled cell volume

Analyse distribution of red blood cell count chronic as well as non chronic

In [35]:
```
grid=sns.FacetGrid(kidney,hue='class',aspect=2)
grid.map(sns.kdeplot,'red blood cell count')
grid.add_legend()
```

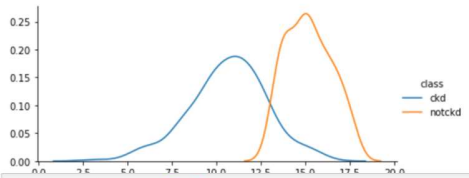
Out[35]: <seaborn.axisgrid.FacetGrid at 0x2777c304580>



from above visuals we can say that person with lower rbc count have high chances of having chronic disease

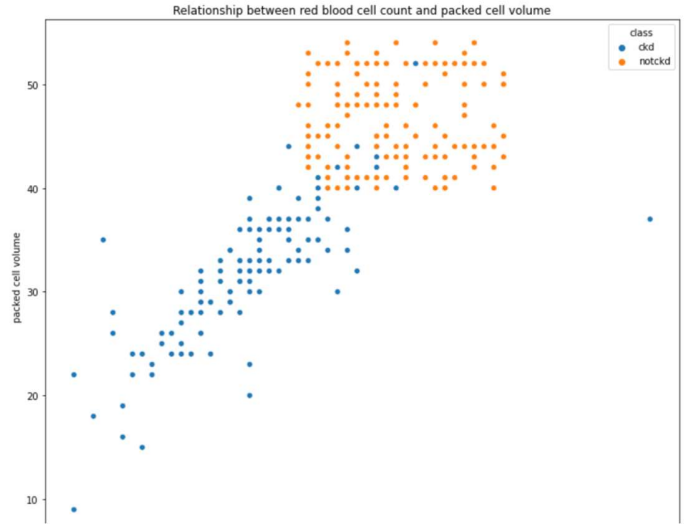
In [36]:
```
grid=sns.FacetGrid(kidney,hue='class',aspect=2)
grid.map(sns.kdeplot,'haemoglobin')
grid.add_legend()
```

Out[36]: <seaborn.axisgrid.FacetGrid at 0x2777c304220>



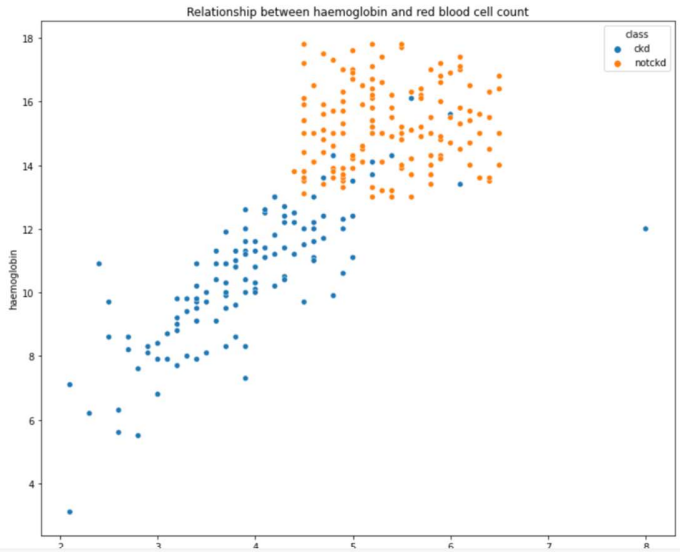
In [37]:
```
plt.figure(figsize=(12,10))
sns.scatterplot(x=kidney['red blood cell count'],y=kidney['packed cell volume'],hue=kidney['class'])
plt.xlabel('red blood cell count')
plt.ylabel('packed cell volume')
plt.title('Relationship between red blood cell count and packed cell volume')
```

Out[37]: Text(0.5, 1.0, 'Relationship between red blood cell count and packed cell volume')



In [38]:
```
plt.figure(figsize=(12,10))
sns.scatterplot(x=kidney['red blood cell count'],y=kidney['haemoglobin'],hue=kidney['class'])
plt.xlabel('red blood cell count')
plt.ylabel('haemoglobin')
plt.title('Relationship between haemoglobin and red blood cell count')
```

Out[38]:
```
Text(0.5, 1.0, 'Relationship between haemoglobin and red blood cell count')
```



Handling Missing Values

In [39]:
```
kidney.isnull().sum()
```

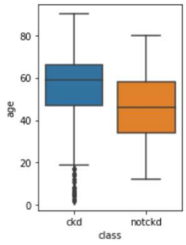
Out[39]:
```
age                          9
blood pressure              12
specific gravity            47
albumin                     46
sugar                       49
red blood cells            152
 pus cell                   65
pus cell clumps              4
bacteria                     4
blood glucose random        44
blood urea                  19
serum creatinine            17
sodium                      87
potassium                   88
haemoglobin                 52
packed cell volume          71
white blood cell count     106
red blood cell count       131
ypertension                  2
diabetes mellitus            2
coronary artery disease      2
appetite                     1
pedal edema                  1
anemia                       1
class                        0
dtype: int64
```

In [40]:
```
kidney.isnull().sum().sort_values(ascending=False)
```

Out[40]:
```
red blood cells            152
red blood cell count       131
white blood cell count     106
potassium                   88
sodium                      87
packed cell volume          71
 pus cell                   65
haemoglobin                 52
```

In [41]:
```
plt.subplot(1,2,1)
sns.boxplot(x=kidney['class'],y=kidney['age'])
```

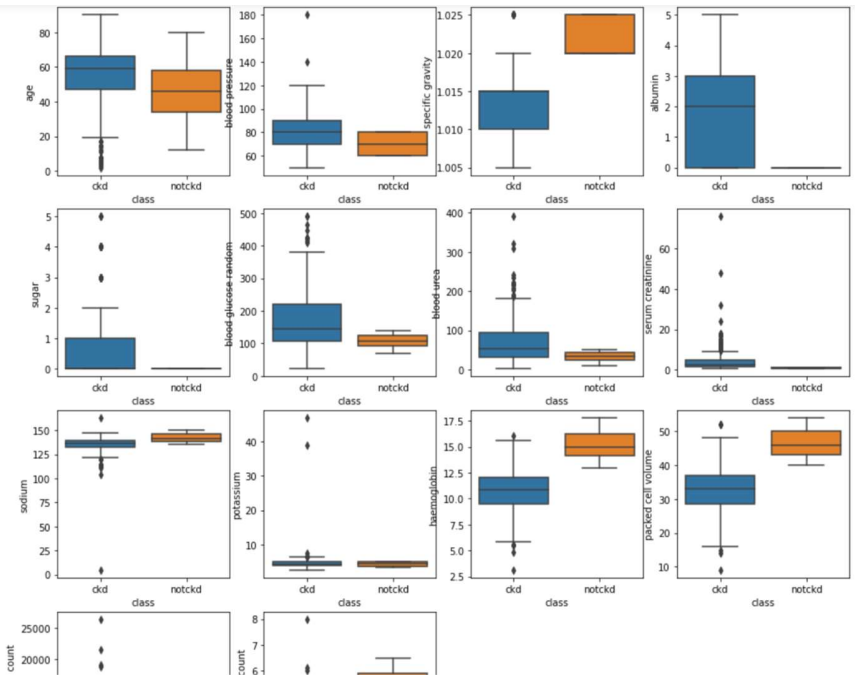
Out[41]:
```
<AxesSubplot:xlabel='class', ylabel='age'>
```



In [42]:
```
list(enumerate(cat_col))
```

Out[42]:
```
[(0, 'red blood cells'),
 (1, ' pus cell'),
 (2, 'pus cell clumps'),
 (3, 'bacteria'),
 (4, 'ypertension'),
 (5, 'diabetes mellitus'),
 (6, 'coronary artery disease'),
 (7, 'appetite'),
 (8, 'pedal edema'),
 (9, 'anemia'),
 (10, 'class')]
```

In [43]:
```
plt.figure(figsize=(15,15))
for i in enumerate(num_col):
    plt.subplot(4,4,i[0]+1)
    sns.boxplot(x=kidney['class'],y=i[1],data=kidney.reset_index())
```



In [44]:
```
np.mean(kidney)
```

Out[44]:
```
age                          51.483376
blood pressure               76.469072
specific gravity              1.017408
albumin                       1.016949
sugar                         0.450142
blood glucose random        148.036517
blood urea                   57.425722
serum creatinine              3.072454
sodium                      137.528754
potassium                     4.627244
haemoglobin                  12.526437
packed cell volume           38.884498
white blood cell count     8406.122449
red blood cell count          4.707435
dtype: float64
```

In [45]:
```
kidney.isnull().sum()
```

Out[45]:
```
age                          9
blood pressure              12
specific gravity            47
albumin                     46
sugar                       49
red blood cells            152
 pus cell                   65
pus cell clumps              4
bacteria                     4
blood glucose random        44
blood urea                  19
serum creatinine            17
sodium                      87
potassium                   88
haemoglobin                 52
packed cell volume          71
white blood cell count     106
red blood cell count       131
ypertension                  2
diabetes mellitus            2
coronary artery disease      2
appetite                     1
```

In [46]:
```
for i in num_col:
    kidney[i].fillna(kidney[i].median(),inplace=True)
```

In [47]:
```
kidney.isnull().sum()
```

Out[47]:
```
age                          0
blood pressure               0
specific gravity             0
albumin                      0
sugar                        0
red blood cells            152
 pus cell                   65
pus cell clumps              4
bacteria                     4
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
ypertension                  2
diabetes mellitus            2
coronary artery disease      2
appetite                     1
pedal edema                  1
anemia                       1
class                        0
dtype: int64
```

In [48]:
```
kidney.describe()
```

Out[48]:

	age	blood pressure	specific gravity	albumin	sugar	blood glucose random	blood urea	serum creatinine	sodium	potassium	haemoglobin	packed cell volume	w
count	400.000000	400.000000	400.000000	400.00000	400.000000	400.000000	400.000000	400.000000	400.000000	400.000000	400.00000	400.000000	4
mean	51.562500	76.575000	1.017712	0.90000	0.395000	145.062500	56.693000	2.997125	137.631250	4.577250	12.54250	39.082500	82

Filling missing values in categorical columns using random values

It was more important to find the missing values and need to clean thos emissing values by using different menthods. (I've dropped the NULL Values). Missing Values leads to False Output and sometimes cause many Problems while Evaluating our Model.

In [49]:
```
kidney['red blood cells'].isnull().sum()
```

Out[49]:
```
152
```

In [50]:
```
random_sample=kidney['red blood cells'].dropna().sample(152)
```

In [51]:
```
random_sample
```

Out[51]:
```
239      normal
369      normal
20     abnormal
147      normal
7        normal
         ...
150      normal
74     abnormal
181      normal
22       normal
254      normal
Name: red blood cells, Length: 152, dtype: object
```

In [52]:
```
kidney[kidney['red blood cells'].isnull()].index
```

Out[52]:
```
Int64Index([  0,   1,   5,   6,  10,  12,  13,  15,  16,  17,
            ...
            245, 268, 280, 290, 295, 309, 322, 349, 350, 381],
           dtype='int64', length=152)
```

In [53]:
```
random_sample.index
```

Out[53]:
```
Int64Index([239, 369,  20, 147,   7, 325, 255, 285, 190, 313,
            ...
```

Performing the Feature Encoding

Machine learning models can only work with numerical values. For this reason, it is necessary to transform the categorical values of the relevant features into numerical ones. This process is called feature encoding.

In [67]:
```
for col in cat_col:
    print('{} has {} categories'.format(col,kidney[col].nunique()))
```

```
red blood cells has 2 categories
 pus cell has 2 categories
pus cell clumps has 2 categories
bacteria has 2 categories
ypertension has 2 categories
diabetes mellitus has 3 categories
coronary artery disease has 2 categories
appetite has 2 categories
pedal edema has 2 categories
anemia has 2 categories
class has 2 categories
```

Label Encoding ---> Because there are less no. of categories in each column

LabelEncoder can be used to normalize labels. It can also be used to transform non-numerical labels (as long as they are hashable and comparable) to numerical labels. Fit label encoder. ¶

- normal -- 0
- abnormal --1

In [68]:
```
from sklearn.preprocessing import LabelEncoder
```

In [69]:
```
le=LabelEncoder()
```

In [70]:
```
for col in cat_col:
    kidney[col]=le.fit_transform(kidney[col])
```

	age	blood pressure	specific gravity	albumin	sugar	red blood cells	pus cell	pus cell clumps	bacteria	blood glucose random	...	packed cell volume	blood cell count	blood cell count	ypertension	diabetes mellitus	coronary artery disease	appetite	pe edi
0	48.0	80.0	1.020	1.0	0.0	1	1	0	0	121.0	...	44.0	7800.0	5.2	1	2	0	0	
1	7.0	50.0	1.020	4.0	0.0	1	1	0	0	121.0	...	38.0	6000.0	4.8	0	1	0	0	
2	62.0	80.0	1.010	2.0	3.0	1	1	0	0	423.0	...	31.0	7500.0	4.8	0	2	0	1	
3	48.0	70.0	1.005	4.0	0.0	1	0	1	0	117.0	...	32.0	6700.0	3.9	1	1	0	1	
4	51.0	80.0	1.010	2.0	0.0	1	1	0	0	106.0	...	35.0	7300.0	4.6	0	1	0	0	

5 rows × 25 columns

Selecting important features

- **SelectKBest**: Feature selection is a technique where we choose those features in our data that contribute most to the target variable. In other words we choose the best predictors for the target variable. The classes in the sklearn.
- **chi2**: A chi-square (χ2) statistic is a test that measures how a model compares to actual observed data. ... The chi-square statistic compares the size any discrepancies between the expected results and the actual results, given the size of the sample and the number of variables in the relationship.

$$chi2.pdf(x,a) = \frac{1}{(2 * gamma\left(\frac{a}{2}\right) * \left(\frac{x}{2}\right)^{\left(\frac{a}{2}-1\right)} * exp\left(\frac{-x}{2}\right))}$$

In [72]:
```
from sklearn.feature_selection import SelectKBest
```

In [73]:
```
from sklearn.feature_selection import chi2
```

In [74]:
```
ind_col=[col for col in kidney.columns if col!='class']
dep_col='class'
```

In [75]:
```
X=kidney[ind_col]
y=kidney[dep_col]
```

In [76]:
```
X.head()
```

Out[76]:

	age	blood pressure	specific gravity	albumin	sugar	red blood cells	pus cell	pus cell clumps	bacteria	blood glucose random	...	haemoglobin	packed cell volume	white blood cell count	red blood cell count	ypertension	diabetes mellitus	coronary artery disease
0	48.0	80.0	1.020	1.0	0.0	1	1	0	0	121.0	...	15.4	44.0	7800.0	5.2	1	2	0
1	7.0	50.0	1.020	4.0	0.0	1	1	0	0	121.0	...	11.3	38.0	6000.0	4.8	0	1	0
2	62.0	80.0	1.010	2.0	3.0	1	1	0	0	423.0	...	9.6	31.0	7500.0	4.8	0	2	0
3	48.0	70.0	1.005	4.0	0.0	1	0	1	0	117.0	...	11.2	32.0	6700.0	3.9	1	1	0
4	51.0	80.0	1.010	2.0	0.0	1	1	0	0	106.0	...	11.6	35.0	7300.0	4.6	0	1	0

5 rows × 24 columns

In [77]:
```
imp_features=SelectKBest(score_func=chi2,k=20)
```

In [78]:
```
imp_features=imp_features.fit(X,y)
```

In [79]:
```
imp_features
```

Out[79]:
```
SelectKBest(k=20, score_func=<function chi2 at 0x000002777BDCE670>)
```

In [80]:
```
imp_features.scores_
```

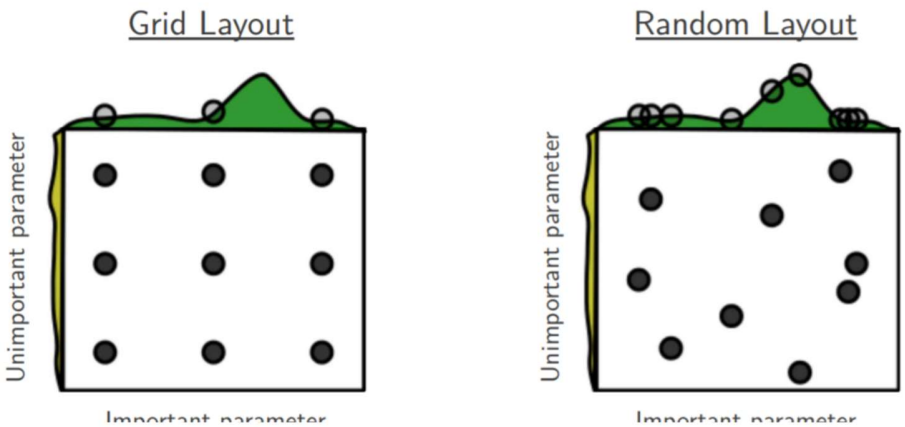
Out[80]:
```
array([1.15859940e+02, 8.17867015e+01, 5.03531613e-03, 2.16000000e+02,
```

Since we are using XGBoost , feature scaling is not required

In [100]:
```
from xgboost import XGBClassifier
```

In [101]:
```
params={'learning-rate':[0,0.5,0.20,0.25],
        'max_depth':[5,8,10],
       'min_child_weight':[1,3,5,7],
       'gamma':[0.0,0.1,0.2,0.4],
       'colsample_bytree':[0.3,0.4,0.7]}
```

RandomizedSearchCV :Randomized search on hyper parameters. RandomizedSearchCV implements a "fit" and a "score" method. It also implements "score_samples", "predict", "predict_proba", "decision_function", "transform" and "inverse_transform" if they are implemented in the estimator used.



Let's Predict our model Accuracy.

In [110]:
```
y_pred=classifier.predict(X_test)
```

In [111]:
```
y_pred
```

Out[111]:
```
array([0, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1,
       0, 1, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0,
       1, 0, 1, 1, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1,
       0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 0, 1,
       1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1,
       0, 0, 1, 0, 1, 0, 1, 0, 0, 1])
```

Evaluation of the model

In [112]:
```
from sklearn.metrics import confusion_matrix,accuracy_score
```

In [113]:
```
confusion_matrix(y_test,y_pred)
```

Out[113]:
```
array([[70,  2],
       [ 0, 48]], dtype=int64)
```

In [114]:
```
accuracy_score(y_test,y_pred)
```

Out[114]:
```
0.9833333333333333
```

> As we Performed all the Methods and Trained our Model using different Menthods
>
> **We Got Very Good Accuracy Using XGBoost - 98% Accuracy**

PLAGIARISM REPORT

B140 Report

ORIGINALITY REPORT

8%	4%	3%	4%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Liverpool John Moores University Student Paper	1%
2	Francisco Javier Campos Zabala. "Chapter 9 Supervised and Unsupervised Learning", Springer Science and Business Media LLC, 2023 Publication	1%
3	Submitted to Coventry University Student Paper	<1%
4	M. Sailaja, G. Lalitha Kumari, Monika Sai Pinninti, Mandava Rupaswi et al. "Crop Yield Prediction Based on Machine Learning	<1%

5	ijarcce.com Internet Source	<1 %
6	Submitted to Roehampton University Student Paper	<1 %
7	Submitted to SRM University Student Paper	<1 %
8	Submitted to University of Wales Institute, Cardiff Student Paper	<1 %
9	Submitted to Westcliff University Student Paper	<1 %
10	www.mdpi.com Internet Source	<1 %
11	Submitted to National College of Ireland Student Paper	<1 %
12	ijritcc.org Internet Source	<1 %

13	Submitted to Berlin School of Business and Innovation Student Paper	<1%
14	Ton Duc Thang University Publication	<1%
15	Submitted to University of North Texas Student Paper	<1%
16	Submitted to King's College Student Paper	<1%
17	Submitted to Manchester Metropolitan University Student Paper	<1%
18	Submitted to Sim University Student Paper	<1%
19	www.sarkisian.ru Internet Source	<1%

20	Submitted to Asia Pacific University College of Technology and Innovation (UCTI) Student Paper	<1 %
21	Submitted to Mahidol University Student Paper	<1 %
22	Submitted to University of Northumbria at Newcastle Student Paper	<1 %
23	digitalcommons.kennesaw.edu Internet Source	<1 %
24	Submitted to CSU, San Jose State University Student Paper	<1 %
25	Submitted to Deakin University Student Paper	<1 %
26	Submitted to Houston Community College Student Paper	<1 %
27	Shani Verma, Shrivishal Tripathi, Anurag Singh, Muneendra Ojha, Ravi R Saxena. "Insect Detection and Identification using YOLO Algorithms on Soybean Crop", TENCON	<1 %

27	Shani Verma, Shrivishal Tripathi, Anurag Singh, Muneendra Ojha, Ravi R Saxena. "Insect Detection and Identification using YOLO Algorithms on Soybean Crop", TENCON	<1%

2021 - 2021 IEEE Region 10 Conference (TENCON), 2021
Publication

28	Submitted to University of Southern Queensland Student Paper	<1%
29	www.relataly.com Internet Source	<1%
30	"Computer Analysis of Images and Patterns", Springer Science and Business Media LLC, 2017 Publication	<1%

31	Submitted to Technological University Dublin Student Paper	<1 %
32	www.ir.juit.ac.in:8080 Internet Source	<1 %
33	baadalsg.inflibnet.ac.in Internet Source	<1 %
34	medium.com Internet Source	<1 %
35	www.ijraset.com Internet Source	<1 %
36	www.medtruth.org Internet Source	<1 %
37	www.researchgate.net Internet Source	

		<1 %
38	www.stat.auckland.ac.nz Internet Source	<1 %
39	www.techtarget.com Internet Source	<1 %
40	Submitted to University of Derby Student Paper	<1 %
41	link.springer.com Internet Source	<1 %
42	dokumen.pub Internet Source	<1 %
43	ijlr.org Internet Source	<1 %
44	journals.plos.org Internet Source	<1 %
45	utpedia.utp.edu.my Internet Source	<1 %

46	Ramesh Chandra Poonia, Mukesh Kumar Gupta, Ibrahim Abunadi, Amani Abdulrahman Albraikan et al. "Intelligent Diagnostic Prediction and Classification Models for Detection of Kidney Disease", Healthcare, 2022 Publication	<1 %

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	Off		

Format - I

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

(Deemed to be University u/s 3 of UGC Act, 1956)

Office of Controller of Examinations

REPORT FOR PLAGIARISM CHECK ON THE DISSERTATION/PROJECT REPORTS FOR UG/PG PROGRAMMES

(To be attached in the dissertation/ project report)

1	Name of the Candidate **(IN BLOCK LETTERS)**	1. PALAK RANI 2. PALAK PATEL
2	Address of the Candidate	Tower 4, Estancia tower ship, Gurvancherry, Potheri, Chennai
3	Registration Number	1. RA2011003010661 2. RA2011003010666
4	Date of Birth	1. 01/02/2003 2. 13/08/2002
5	Department	Computer Science and Engineering
6	Faculty	Engineering and Technology, School of Computing
7	Title of the Dissertation/Project	CHRONIC KIDNEY DISEASE PREDICTION
8	Whether the above project /dissertation is done by	~~Individual~~ or group : (Strike whichever is not applicable) **a)** If the project/ dissertation is done in group, then how many students together completed the project : 2 b) Mention the Name & Register number of other candidates :
9	Name and address of the Supervisor / Guide	Dr. S.Babu Assistant Professor Department of Computing Technologies SRM Nagar, Kattankulathur - 603 203 Chengalpattu District, Tamil Nadu **Mail ID:** babus@srmist.edu.in **Mobile Number:**9894774787
10	Name and address of Co-Supervisor / Co- Guide (if any)	**NIL**

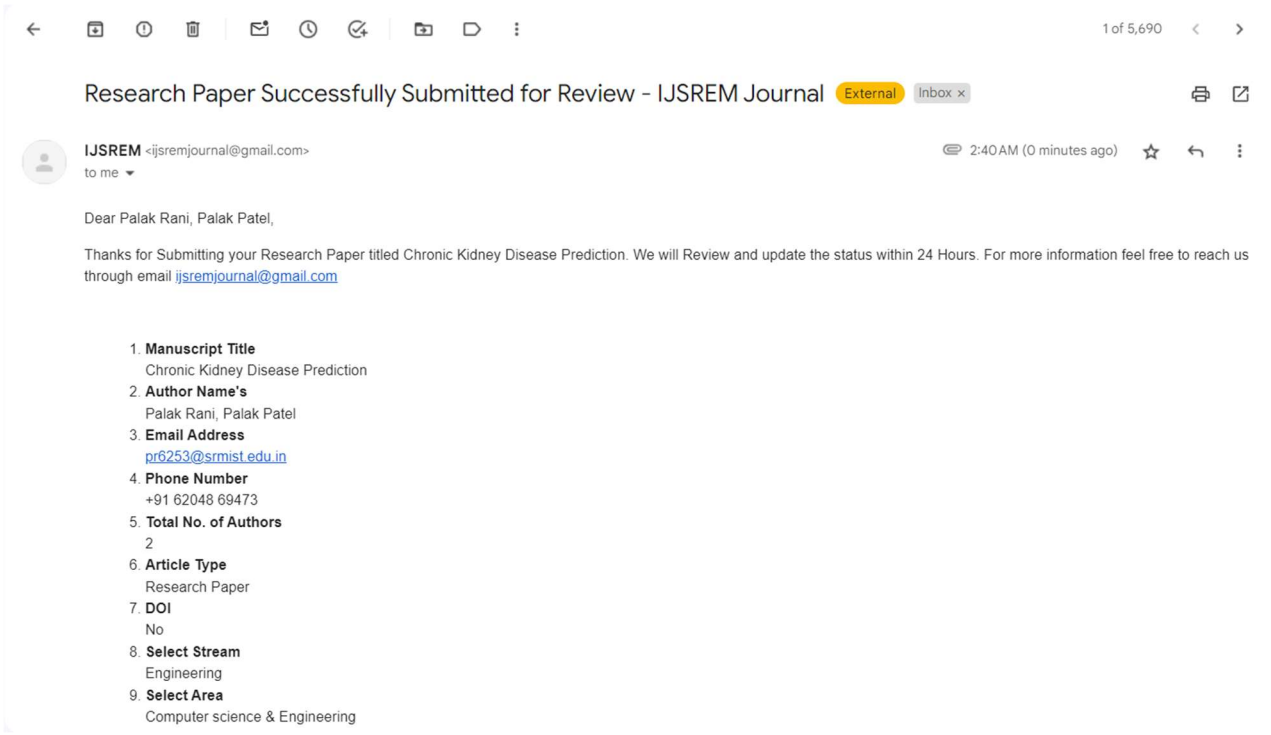
11	Software Used	Turnitin		
12	Date of Verification			
13	**Plagiarism Details: (to attach the final report from the software)**			
Chapter	**Title of the Chapter**	**Percentage of similarity index (including self citation)**	**Percentage of similarity index (Excluding self-citation)**	**% of plagiarism after excluding Quotes, Bibliography, etc.,**
1	Introduction	1	1	1
2	Literature Survey	2	2	2
3	Modules and Methodology	0	0	0
4	Result & Discussion	3	3	3
5	Conclusion & Future Enhancement	0	0	0
Appendices		1	1	1

I / We declare that the above information have been verified and found true to the best of my / our knowledge.

Signature of the Candidate	**Name & Signature of the Staff (Who uses the plagiarism check software)**
Name & Signature of the Supervisor/ Guide	**Name & Signature of the Co-Supervisor/Co-Guide**

Name & Signature of the HOD

PAPER PUBLICATION STATUS



1 of 5,690

Research Paper Successfully Submitted for Review - IJSREM Journal External Inbox ×

IJSREM <ijsremjournal@gmail.com> 2:40 AM (0 minutes ago)
to me

Dear Palak Rani, Palak Patel,

Thanks for Submitting your Research Paper titled Chronic Kidney Disease Prediction. We will Review and update the status within 24 Hours. For more information feel free to reach us through email ijsremjournal@gmail.com

1. **Manuscript Title**
   Chronic Kidney Disease Prediction
2. **Author Name's**
   Palak Rani, Palak Patel
3. **Email Address**
   pr6253@srmist.edu.in
4. **Phone Number**
   +91 62048 69473
5. **Total No. of Authors**
   2
6. **Article Type**
   Research Paper
7. **DOI**
   No
8. **Select Stream**
   Engineering
9. **Select Area**
   Computer science & Engineering