Chronic kidney disease prognosis using Artificial Intelligence

Submitted as Research Report Honours in SIT724

SUBMISSION DATE
T1-2024

Sourav Sharma
STUDENT ID 220080187
COURSE - Bachelor of Software Engineering Honours (S464)

Supervised by: Prof, Daniel Lai

Abstract

Chronic Kidney Disease (CKD) presents an increasing global health challenge, necessitating advanced predictive models for timely interventions. This thesis undertakes a comprehensive evaluation of machine learning algorithms aimed at predicting CKD incidence as well as kidney failure using temporal data, with a focus on Light Gradient Boosting Machine (LightGBM), Random Forest, Decision Tree, and Extreme Gradient Boosting (XGBoost) algorithms. Employing a robust experimental design, the study used two diverse datasets, one from Japan and another from UCI, each pre-processed to develop kidney failure prediction models and CKD detection models respectively. The developed models include the hyper-parameter models, while their performances were recorded and code made available for further research. The results showed a remarkable advantage for LightGBM in CKD detection with F1 score of 98% whereas XGBM was the best model in kideny failure prediciton (F1 score of 98.00%). The thesis also includes a comparative analysis and identifies opportunities for future work. In the course of this investigation, a notable pattern of consistently high performance when temporal values were incorporated into the kidney failure prediction model was observed. This suggests that the time-based features play a pivotal role in enhancing the model's predictive accuracy.

Acknowledgments

I extend my sincerest gratitude to my supervisor, Dr. Daniel Lai, for her invaluable guidance and support throughout the course of this thesis.

Contents

1 Introduction 1
- 1.1 Structure of thesis . . . 1

2 Literature Review 3
- 2.1 Introduction . . . 3
- 2.2 Background . . . 5
  - 2.2.1 Definitions and Classifications Linked with CKD . . . 6
  - 2.2.2 Introduction of AI in CKD prognosis . . . 7
  - 2.2.3 Kidney Failure Risk Equation (KFRE) . . . 8
- 2.3 Previous work . . . 9
- 2.4 Research Gap . . . 10
- 2.5 Research Questions . . . 11
  - 2.5.1 Feature Selection in CKD Prognosis . . . 11
  - 2.5.2 Integration of Temporal Data . . . 11
  - 2.5.3 Addressing Data Quality Issues . . . 11
- 2.6 Summary . . . 12

3 Research Design & Methodology 12
- 3.1 Introduction . . . 12
- 3.2 Model Implementation . . . 13
- 3.3 Research design . . . 15
- 3.4 Part 1 - Early Detection of CKD . . . 15
  - 3.4.1 Core Problem . . . 16
  - 3.4.2 Methodology . . . 16
- 3.5 Part 2 - Kidney Failure Prediction in CKD patients . . . 22
  - 3.5.1 Core Problem . . . 22
  - 3.5.2 Methodology . . . 22
- 3.6 Performance evaluation . . . 27
- 3.7 Conclusion . . . 30

4 Artefact Development Approach 30

5 Empirical Evaluation 30
- 5.1 Introduction . . . 30
- 5.2 Model Implementation . . . 31
- 5.3 Performance evaluation . . . 31
- 5.4 Part 1 - Early Detection of CKD . . . 33
  - 5.4.1 Core Problem . . . 33
  - 5.4.2 Environment and code availability . . . 33
  - 5.4.3 Data and Results . . . 34

6 Results & Discussion 43
- 6.1 Novelty of the Study . . . 43
- 6.2 Comparative Analysis and Validation . . . 44
- 6.3 Limitations . . . 44

7 Conclusion & Future Work 44
7.1 Future works 45

1 Introduction

The alarming rise in the prevalence of Chronic Kidney Disease (CKD), affecting approximately 850 million people worldwide, punctuates an urgent need for novel intervention strategies [1]. Ranked as the 18th most lethal disease as of 2010, CKD not only threatens human life but also imposes a staggering economic burden [2][3]. Specifically, medicare spending in the United States reached $20,432 per adult aged 65 or above in 2011, spotlighting the economic gravity of this health crisis [3]. CKD can affect various parts of the body, leading to several potential complications such as weak bones, hyperparathyroidism, and brain dysfunction [4]. Moreover, the fact that it is incurable and remains to go unnoticed until later stages makes it more dangerous[2]. According to official guidelines, CKD is divided majorly into five stages based on the value of a bio-marker called estimated glomerular filtration rate (eGFR). The value of eGFR is indicative of kidney function, therefore, lower the value of eGFR indicates lower kidney function[4]. At fifth stage of CKD, kidney function stops completely (can be refered as kidney failure) and the patients require external techniques/tools such as dialysis for survival. Several studies have been conducted to diagnose the CKD patients using both AI and non-AI methods, however, few studies have explored the prognosis of CKD using AI. In particular, the topic of kidney failure predictions using AI remains largely unexplored.

This study aims to explore CKD prognosis using AI and will advance from broader CKD diagnosis methods to development of more specific topics such as kidney failure prediction models using AI. The study, in particular, explores the impact of temporal data in predicting the kidney failure instances among CKD patients. Building a generalized kidney failure prediction model is also an area of interest for this study. However, the actual implementation and development of the generalized model for kidney failure prediction is planned for the future.

1.1 Structure of thesis

The first chapter of the thesis details the general introduction of the CKD and the study structure. Chapter 2 serves as the Literature Review, providing essential background information, an overview of previous research, and pinpointing research gaps. The third chapter details the Research Design and Methodology, which outlines the experimental framework, research paradigms, and ethical considerations for the study. This is followed by chapter 4, the Artefact Development Approach, which details the information on the model implementation and feature engineering processes. Chapter 5, termed as Empirical Evaluation, offers rigorous testing and comparative analysis of various AI models employed in the study. The subsequent chapter 6, Results & Discussion, focuses on interpreting the study's findings and their implications. The thesis concludes with chapter 7, which encapsulates the research and outlines potential avenues for future work.

2 Literature Review

2.1 Introduction

The kidneys are essential organs responsible for regulating salt, potassium, and various harmful substances in the human body, managing approximately 5 liters of blood. When kidney function deteriorates, waste products accumulate in the bloodstream, transforming this blood into 9-10 liters of a toxic fluid containing urea and creatinine within 2-3 days. This condition is known as chronic kidney disease (CKD) [5].

Chronic kidney disease impacts ten percent of people worldwide, presenting a major health obstacle because of its painful and life-threatening effects as it progresses. In 2010, CKD was ranked as the 18th most common cause of death globally [6]. Approximately 850 million people are affected by this disease globally, making it a significant public health concern. CKD encompasses a broad range of heterogeneous disorders affecting general functions and the state of the kidney over an extended period [7]. This accumulates over a long period, robbing the kidneys of their ability to filter blood, removing waste, and maintaining fluids.

One factor which is helpful in describing the severity of the disease is its economic burden around the globe. US reports have recorded an annual Medicare spending of $20,432 per person for adults aged 65 years or above in 2011 [8]. Additionally, CKD has a significant economic impact on healthcare systems, particularly in low-to-middle-income countries where a large number of individuals die because they cannot access affordable treatments. Ageing populations and the increasing prevalence of Type 2 diabetes worsen the seriousness of CKD [8]. In general, chronic kidney disease affects social, economic, and psychological aspects of an individual's life. For instance, the later stages of the disease may impact mental and behavioural aspects of the patients [9].

Between the years 2011 and 2012, around 1.9 million adults in England were recorded as having CKD in the Quality Outcomes Framework (QOF). The estimated percentage of people in England with stages G3-G5 CKD is 6.8%, which equates to approximately

3 million individuals. It is estimated that each CKD patient recorded in the QOF costs the NHS around £795 annually, resulting in a total expenditure of about £1.44 to £1.45 billion every year.

In its early stages, CKD is often asymptomatic, making testing the only method to diagnose the condition. Early detection is crucial, as it allows for effective treatment that can prevent progression to end-stage renal disease (ESRD) [8, 7]. Kidney Disease: Improving Global Outcomes (KDIGO), an organization that provides guidelines on the diagnosis and management of global kidney diseases, stratified CKD into five different stages (G1-G5), with each successive stage exhibiting worsening kidney functions measured by a biomarker known as glomerular filtration rate (GFR) [6]. Continuous progression of chronic kidney disease leads to end-stage renal disease (ESRD), also indicated as G5 in KDIGO guidelines, where the kidney effectively loses all its functionalities leading to waste and fluid retention [7]. This necessitates the implementation of kidney replacement therapy (KRT) such as dialysis or kidney transplant [5]. Such treatments are expensive and afterwards require constant care of the patients. For instance, a study described the average cost for the first year following a kidney transplant to be approximately $75,326, as per data standardized to 2022 U.S. dollars and accounting for Purchasing Power Parity.

Early treatment of chronic kidney disease can help in either stopping or slowing down the progression of the disease towards end-stage renal disease, and thus, early prognosis will provide greater assistance in combating the disease and reduce mortality rate [7, 10]. It is recommended that individuals with CKD risk factors, such as a family history of kidney failure, hypertension, or diabetes, undergo annual screenings. Early detection and treatment of CKD have the potential to slow or halt the progression of the disease [5, 8, 7, 10, 9]. In addition, studying chronic kidney disease progression has enabled scientists to identify correlations between different factors from comorbid diseases to blood biomarkers, as well as improving the existing methods of handling CKD patients. For example, Major et al. [11] proposed a method to triage CKD referrals from primary to secondary care using kidney-failure risk equation and urine albumin-to-creatinine ratio, thus improving not only the existing handling framework but also reducing otherwise unnecessary expenditure on medical procedures.

In this context, computer-aided diagnosis can be instrumental in the early and efficient prognosis of CKD. Machine learning (ML), a subfield of artificial intelligence (AI), can be utilized for the precise identification of the disease. These systems are designed to assist clinical decision-makers in achieving more accurate disease classification, thereby enhancing diagnostic accuracy and improving patient outcomes [5, 10]. The application of AI algorithms reduces the likelihood of human errors and enhances predictive accuracy [10, 9]. The effective implementation of these algorithms facilitates early disease detection, thereby decreasing mortality rates and enabling timely treatment for patients [5, 10].

The field of research in CKD has been continually expanding, with many researchers focusing on several aspects. Investigating the correlation between CKD patients with comorbid diseases and kidney failure, progression of CKD across different genders and age groups, and examining CKD as a contributing factor for COVID-19 mortality are a few areas of interest among others [11]. However, numerous researchers have shown interest in implementing machine learning approaches for the development of CKD prediction models. Thus, the topic of interest for this literature review is chronic kidney disease prognosis to kidney failure using artificial intelligence (AI)/machine learning (ML) techniques.

This literature review aims to explore the prediction approaches used in CKD prognosis, focusing on AI and ML techniques, and predicting the risk of complete kidney failure or ESRD in CKD patients. The review is structured into five main sections: Section 1 provides the introduction; Section 2 discusses the background, including definitions and related terminology linked with kidney failure; Section 3 reviews related work, providing insights into existing studies and methodologies; Section 4 identifies research gaps and presents relevant research questions; and Section 5 provides the summary, synthesizing the findings and highlighting potential future research directions.

2.2 Background

In order to analyze previous and current research, it is essential to comprehend important technical terminology and instruments. This part gives a precise explanation of CKD, presents the various stages as categorized by official organizations, explores the reasoning behind utilizing AI/ML in CKD prognosis, and gives a summary of the basic methods used for forecasting kidney failure.

2.2.1 Definitions and Classifications Linked with CKD

In medical terminology, chronic kidney disease (CKD) is characterized by either kidney damage (indicated by the presence of albuminuria) or reduced kidney function (marked by a glomerular filtration rate below 60 mL/min per 1.73 m^2) persisting for three months or longer [3]. Due to the challenge of directly measuring the glomerular filtration rate, healthcare professionals rely on estimated glomerular filtration rate (eGFR) to monitor the progression of the disease [6]. The KDIGO work group provides comprehensive guidelines on the definition, classification, disease prognosis, complication management, patient care, and treatment protocols for CKD [3].

Stage	Description	Estimated Glomerular Filtration Rate (eGFR)
1	Decreased kidney functions	> 90 mL/min/1.73m^2
2	Mild loss of kidney function	$60 - 89$ mL/min/1.73m^2
3	Moderate loss of kidney function	$30 - 59$ mL/min/1.73m^2
4	Severe loss of kidney function	$15 - 29$ mL/min/1.73m^2
5	Complete kidney failure	< 15 mL/min/1.73m^2

Table 1: Different stages of CKD progression [12]

Table 1 presents the various stages of CKD progression as defined by the National Kidney Foundation Outcomes Quality Initiative (NKF-K/DOQI). These stages range from stage 1, indicating decreased kidney function, to stage 5, which signifies complete kidney failure or ESRD, characterized by a glomerular filtration rate of less than 15 mL/min per 1.73m^2 [12, 4].

It is suggested that the classification system proposed by the Kidney Disease: Improving Global Outcomes (KDIGO) work group may provide a more precise characterization of patient prognosis. This system includes additional criteria such as albuminuria and urinary albumin loss [12]. Table 2 details the KDIGO categories based on urinary albumin loss.

Despite the implementation of the aforementioned guidelines in practical scenarios, they have notable limitations. One significant limitation is that these guidelines are only applicable to children under two years of age. Furthermore, they emphasize

Category	Albuminuria 24 hours	Albumin/creatinine rate	Description
A1	> 30 mg	> 30 mg/g	Normal or moderately elevated
A2	$30 - 300$ mg	$30 - 300$ mg/g	Moderately high
A3	> 300 mg	> 300 mg/g	Severely elevated

Table 2: Different categories of albuminuria in CKD [12]

albuminuria while neglecting age and gender-related factors [12].

With advancements in technology and modern techniques, researchers have generated vast amounts of data. However, the proportion of meaningful data extracted remains relatively small. While the availability of extensive data provides an opportunity to uncover more comprehensive and valuable insights, managing and processing such large datasets pose significant challenges. Therefore, more dynamic, data-driven approaches are necessary.

This context highlights the role of artificial intelligence (AI), which excels in pattern recognition and data-driven tasks [13]. The following section introduces the necessity and advantages of AI in the prognosis of CKD.

2.2.2 Introduction of AI in CKD prognosis

The integration of artificial intelligence (AI) in chronic kidney disease (CKD) prognosis has significantly advanced the field of medical diagnostics. Traditional methods for CKD diagnosis relied heavily on various statistical tests such as blood tests, urine tests, X-rays, glomerular filtration rate (GFR) tests, and kidney scans or biopsies [14]. These approaches, while foundational, often involved manual selection of features to construct predictive models, which could be both time-consuming and prone to human error [14].

The advent of AI has revolutionised these processes by automating data handling and enhancing the predictive power of models. AI techniques have enabled the extraction of meaningful insights from large datasets, identifying complex patterns and underlying fluctuations that traditional methods might miss [15]. This capability is particularly valuable in medical applications where data can be highly non-linear and multifaceted [16]. AI's efficiency in processing extensive and intricate data sets surpasses traditional statistical models, offering promising solutions that are more accurate and less labor-intensive [17].

Early research has demonstrated the potential of AI in improving CKD prognosis.

For instance, Aljaaf et al. [18] evaluated various machine learning models, including Multilayer Perceptron (MLP) and Logistic Regression (LOGR), achieving notable performance improvements in CKD detection. These models benefitted from advanced data preprocessing techniques, such as multiple imputations and correlation analysis, which refined the input data and enhanced model accuracy.

Feature selection has been identified as a crucial factor in enhancing model performance. Almasoud and Ward [19] explored the efficacy of machine learning algorithms using a minimal set of predictors for CKD detection. They employed statistical tests like ANOVA, Pearson's correlation, and Cramer's V test to eliminate redundant features, achieving high accuracy with Gradient Boosting classifiers. This approach underscored the potential for accurate CKD detection using fewer clinical tests, which is particularly beneficial in resource-limited settings where the cost and accessibility of medical tests are significant concerns.

AI and machine learning (ML) have also shown promise in addressing the multifactorial nature of CKD. Various studies have highlighted the importance of understanding associations between diverse factors such as diabetes, hypertension, age, and ethnicity, which are known to contribute to CKD [20, 21]. AI and ML are adept at handling these complex relationships within datasets, facilitating more robust and precise prognostic models [14]. For example, the development of methods like the kidney failure risk equation illustrates how AI can be utilized to identify key factors for CKD progression and enhance prognosis tools due to its capability to learn and adapt across different scales.

2.2.3 Kidney Failure Risk Equation (KFRE)

Before dwelling into the application of AI for prognosis of CKD to kidney failure, it is important to understand the method for the prognosis. This emphasizes the introduction of kidney failure risk equation (KFRE). The equation is generally considered as the best tool for predicting risk of progression of CKD to kidney failure and has been externally validated by various studies internationally. It uses – 4, 6, or 8 variables depending on the version [13]. Four variable KFRE include age, sex, estimated glomerular filtration rate (eGFR), and urine albumin/creatinine ratio [13].

Even though the equation provides great insight into the kidney failure predictions it is important to underscore its limitations as well. KFRE is a static equation meaning that it focuses only on the current values to predict the outcome over the next 2 or 5 years. It does not take the time-series changes into consideration for the kidney failure prediction. This aspect of the equation can lead to drawbacks in certain cases, for instance, KFRE will indicate very low risk of dialysis or kidney failure even in a patient (aged around 50 years) experiencing a rapid eGFR decline from 82 mL/min/1.73m^2 to around 40 mL/min/1.73m^2 within five years [22]. This increased the risk for the patient as the optimal period for treatment was lost. It can, thereby, be said that inclusion of time-series changes may provide great assistance in accurate prognosis.

2.3 Previous work

This section of the review outlines the previous work done in the field of detection and progression of CKD.

Dingwei et al. [23] serves as a foundational work that employs multivariate logistic regression to identify patients with stages 3 or 4 CKD who are at high risk of progressing to kidney failure within 24 months. The model demonstrates a satisfactory level of accuracy with an AUROC of 84.4%, offering a reliable mechanism for early intervention. Tangri et al. [24] complements this study by advocating for the integration of risk prediction models into the clinical workflow, emphasizing the practicality of these tools in real-world healthcare settings. Chennareddy et al. [25] leveraged machine learning classifiers such as XGBoost, Decision Trees, SVM, and KNN for CKD diagnosis. Notably, XGBoost emerged as the superior model with a 99% accuracy rate, paving the way for more complex and precise methodologies. Arumugham et al. [26] further validated this with a deep neural network model achieving an accuracy rate of 98.75%, providing more intuitive insights into early-stage CKD diagnosis.

On the prognostic front, Yuan et al. [27] broke new ground by merging latent longitudinal trajectory clustering with survival analysis. This nuanced approach helps identify distinct patient subgroups based on their kidney function trajectories, offering personalized care and treatment optimization. Patel et al. [28] refined prognostic methods by applying various machine learning classifiers, with XGBoost standing out with a 99% accuracy rate. Wainstein et al. [13] compared a random forest model and the 4-variable KFRE (2-year) to predict kidney failure. Their study on a cohort of 1365 patients with CKD showed the random forest models outperformed the KFRE with AUCROC ranging from 97.0 to 98.0, demonstrating higher specificity and sensitivity.

Furthermore, Ferguson et al. [29] developed and externally validated a random forest model for predicting CKD progression, achieving an AUC of 0.88 at 2 years and 0.84 at 5 years. Their study involved a large cohort from two Canadian provinces, emphasizing the model's potential for early identification of high-risk patients using routinely collected laboratory data. Whereas, Wang et al. [30] focused on predicting progression to ESRD in Type 2 Diabetes Mellitus patients. Using a robust XGBoost model trained on data from 19,159 patients, they achieved an AUROC of 0.95, highlighting the integration of predictive algorithms into electronic medical records for improved clinical outcomes.

As research advanced, different machine learning models were tested for CKD detection with varying success. Rahat et al. [10] assessed a combined model of XGBoost, Random Forest, Logistic Regression, and AdaBoost, achieving a 95% accuracy, showcasing the benefits of hybrid models. Swain et al. [5] found that SVM attained a testing accuracy of 99.33%, surpassing the Random Forest model's accuracy of 98.67%, making it a strong candidate for CKD classification. The emphasis on hybrid models and ensemble techniques reflects a broader trend in machine learning to improve predictive accuracy and reliability. Aljaaf et al. [8] found that MLP and LOGR models exhibited the best performance, with MLP achieving an AUC of 0.995 and high sensitivity. Almasoud and Ward [7] highlighted the superior performance of Gradient Boosting with an F1-measure of 99.1%, sensitivity of 98.8%, and specificity of 99.3%, underscoring the importance of feature selection.

Recent developments have focused on enhancing model interpretability and reliability. Sawhney et al. [6] demonstrated that Deep Neural Networks (DNN) surpassed other models, achieving perfect accuracy in CKD categorization due to their ability to handle non-linearities and complex data patterns. Rashed-Al-Mahfuz et al. [9] emphasized the high accuracy of the Random Forest model with fewer features, achieving a classification accuracy of 99.50%. This highlights the potential for accurate diagnoses with fewer clinical tests, which is crucial in resource-limited settings.

2.4 Research Gap

While AI/ML have been implemented in the prognosis of CKD, the existing literature is lacking in addressing the role of temporal changes in the determining attributes for CKD progression to kidney failure. Therefore, the following research gaps have been identified in studies aimed at predicting kidney failure using AI:

1. Effective Feature Selection: The selection of relevant features remains a significant challenge, as it directly impacts model performance and efficiency. There is a need for methods to identify and utilize the most predictive features for CKD progression.
2. Inclusion of Temporal Data: Most existing studies lack the integration of temporal or longitudinal data, which is crucial for accurately predicting the occurrence of kidney failure. Incorporating temporal changes in patient data can provide more accurate predictions of disease progression.
3. Data Quality Issues: The presence of small dataset sizes, class imbalances, and missing data continues to be a major issue. These factors can skew model outcomes, reduce generalizability, and hinder the development of reliable predictive models. Effective strategies to address these data quality issues are essential for improving the performance of AI/ML models in CKD prognosis.

2.5 Research Questions

Based on the identified research gaps, the following research questions are proposed:

2.5.1 Feature Selection in CKD Prognosis

1. What are the most effective methods for selecting predictive features in CKD prognosis models, and how do these methods impact the accuracy and efficiency of the models?

2.5.2 Integration of Temporal Data

1. How does the inclusion of temporal or longitudinal data improve the accuracy of predictive models for CKD progression to kidney failure?

2.5.3 Addressing Data Quality Issues

1. What strategies can be employed to mitigate the effects of small dataset sizes, class imbalances, and missing data in CKD prognosis models?

These research questions aim to address the critical challenges identified in the literature and contribute to the development of more accurate and reliable AI/ML models for CKD prognosis.

2.6 Summary

This literature review provides a comprehensive overview of the application of artificial intelligence (AI) in chronic kidney disease (CKD) prognosis. It highlights a significant shift towards AI-driven methodologies for detecting and predicting CKD progression, demonstrating improved diagnostic accuracy and predictive power. A key focus is the potential of AI techniques in predicting the risk of complete kidney failure or end-stage renal disease (ESRD) in CKD patients. The review identifies notable research gaps, particularly the need for incorporating temporal data and effective feature selection in AI models. Additionally, it underscores the importance of publicly available datasets for future research and addresses the substantial economic burden of CKD, emphasizing AI's potential to reduce healthcare costs through early detection. The review concludes by posing relevant research questions aimed at enhancing the accuracy and reliability of AI/ML models for CKD prognosis, thus providing a solid foundation for future investigations in this critical field.

3 Research Design & Methodology

3.1 Introduction

Chronic kidney disease (CKD) is considered as one of the most dangerous diseases in the world as can be demonstrated by its prevalence in approximately 850 million people worldwide [1]. In general, CKD refers to the phenomenon of gradual loss of kidneys in filtering the waste from the blood over an extended period. The project explores the application of machine learning algorithms to develop a predictive model for early prediction of CKD as well as for the occurrence of kidney failure in CKD patients. This report describes the core problem, scope, development details as well as evaluation of proposed solution.

3.2 Model Implementation

In the medical field, interpretability is given greater importance than the black-box models as in a real-life scenario a medical practitioner requires to understand the reasoning behind the models prediction to give more suitable advise [31]. Thus the study utilises supervised machine learning based algorithms than the deep learning. The following models are used in the study:

1. Decision Tree (DT): Decision tree is a supervised machine learning algorithm that is suitable for both classification as well as regression tasks. It includes a root node, branches as well as leaf nodes, and operates by recursively partitioning the feature space into subsets based on the value of input features, creating a tree-like model of decisions. Each internal node of the tree represents a ”test” on a feature, each branch represents the outcome of the test, and each leaf node represents a class label (in classification) or a continuous value (in regression) [32]. Training sets are employed to generate these trees, which are highly interpretable but prone to overfitting if not pruned effectively. Decision Trees easily understandable yet less generalizable compared to ensemble methods [25].

2. Random Forest (RF): Random Forest is an ensemble algorithm that synthesizes the predictive capabilities of multiple decision trees to yield more robust results. Unique in its random selection of features for each tree, the algorithm aims to build an ensemble of decorrelated trees, thereby circumventing the overfitting common in single decision trees. Each internal node represents a feature, and branching occurs based on decision rules determined by the algorithm. Random Forest can be employed in both classification and regression tasks, offering a balanced blend of accuracy and model interpretability [29][33]. Hyper parameters for the decision tree and random forest are similar, and is calculated at each tree. For the random forest, the feature importance is given by the equation 1 [34]:

$$\text{Feature Importance in RF } (X_i) = \frac{1}{T}\sum_{t=1}^{T}\Delta_i(t) \tag{1}$$

   where Feature Importance(X_i) is the importance of feature X_i, T is the total number of trees in the forest, and $\Delta_i(t)$ is the importance of feature X_i in tree t. The importance $\Delta_i(t)$ is typically calculated as the total reduction in the criterion (e.g., Gini impurity or entropy) brought by that feature in the tree.

3. Logistic Regression (LR): Logistic regression is a supervised machine learning algorithm that is widely used for binary classification problems [32]. It is based on linear regression principles, The fundamental concept behind logistic regression is the logistic function, also known as the sigmoid function, which maps any real-valued number into a value between 0 and 1. This property makes logistic regression particularly suitable for modeling the probability of a binary event occurring [34]. The logistic regression model estimates the probability that a given input belongs to a particular class using the equation 2 [35]:

$$P(Y=1|X) = \frac{1}{1+e^{-(\beta_0+\beta_1 X_1+\beta_2 X_2+\ldots+\beta_n X_n)}} \tag{2}$$

   where $P(Y=1|X)$ is the probability of the target variable Y being 1 given the input features X, β_0 is the intercept term, and $\beta_1, \beta_2, \ldots, \beta_n$ are the coefficients corresponding to the input features $X_1, X_2, \ldots, X_n$.

4. Extreme gradient boosting machine (XGBoost): XGBoost, an abbreviation for Extreme Gradient Boosting, serves as an optimized distributed gradient boosting library aimed to be highly efficient, flexible, and portable. Originating as an extension to gradient-boosted decision trees, XGBoost is engineered with computational efficiency and model performance in mind. The algorithm employs the gradient descent algorithm to minimize the loss function, making it particularly effective for a variety of machine learning tasks. Originally developed with support for Python and R, the growing popularity of XGBoost has led to its implementation in multiple programming languages. Its versatility is evident in its application across various domains such as regression, classification, and ranking [33][36]. Originating as an extension to gradient-boosted decision trees, XGBoost is engineered with computational efficiency and model performance in mind [33][36].

   The objective function that XGBoost aims to minimize can be written as[37]:

$$\text{Obj}(\Theta) = \sum_{i=1}^{n} L(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k) \tag{3}$$

   Where,

   Θ : The set of parameters for the base learners.

   $L(y_i, \hat{y}_i)$: The loss function, comparing the actual label y_i with the predicted label $\hat{y}_i$ for each observation i.

n : The total number of observations in the dataset.

$\Omega(f_k)$: The regularization term for each tree f_k.

K : The total number of base learners (trees).

f_k : The k-th base learner (tree) in the ensemble.

For this study, the objective function is 'binary:logistic,' indicating that the problem at hand is one of binary classification.

5. Light gradient boosting machine (LightGB): LightGBM, standing for Light Gradient Boosting Machine, is another ensemble learning technique that focuses on high computational efficiency and memory optimization. Unlike traditional boosting algorithms that employ column-wise histogram techniques, LightGBM utilizes Gradient-based One-Side Sampling (GOSS) to prioritize the sampling of data instances that contribute most to information gain. Initially tailored for high-dimensional large datasets, LightGBM's architecture minimizes memory usage while maximizing computational speed [36].

3.3 Research design

The study takes Quantitative research approach and employs experimental investigation. Additionally, decision was made to undertake Supervised machine learning approach due to experimental nature of the study that provides more control over the project [38]. Various models used in the study are independent variables whereas performance metrics of the models is dependent variable [38]. In clinical environment, understanding the output and underlying principle resulting in that output plays vital role [31]. Thus, this study adopted approaches that place emphasis on interpretablility.

3.4 Part 1 - Early Detection of CKD

3.4.1 Core Problem

As mentioned previously, CKD affects millions of people worldwide, and currently, there are no definitive indicators that can predict CKD. Moreover, CKD is incurable, and its presence often goes unnoticed until the later stages when symptoms become severe [2]. Therefore, early detection is critical for timely intervention and management of the disease [39]. By using the clinical dataset, the study aims to develop several machine learning models to predict the whether the patient is likely to develop chronic kidney disease or not.

3.4.2 Methodology

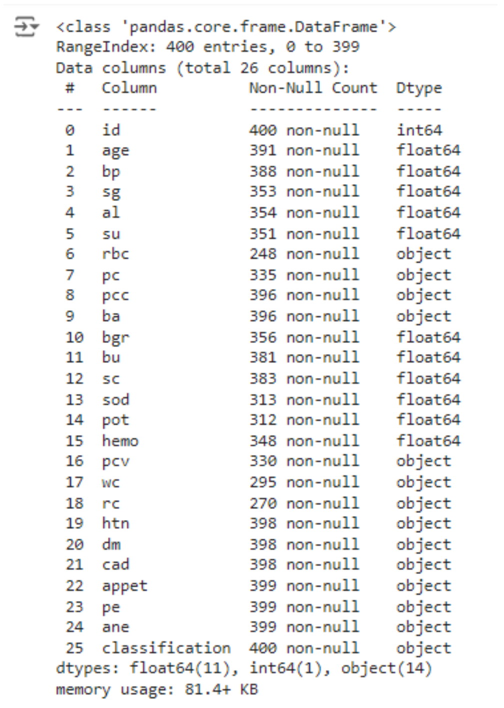
There are main requirements for replicating the study outcomes as described below:

1. Data: The dataset for the study is obtained from University of California Irvine (UCI) Machine Learning Repository available online at https://tinyurl.com/UCI-dataset-public. The dataset contained clinical readings of 400 patients collected from the Apollo Hospital, Karaikudi, India during a nearly 2-month period in 2015. With total number of 25 attributes in total with 11 numeric, 13 nominal features, and 1 target feature (notCKD/CKD) as described in table 6.

Attribute	Name	Category
age	Age [age in years]	Numerical
bp	Blood pressure [diastolic blood pressure in mm/Hg]	Numerical
sg	Specific gravity [specific gravity to compare the density of urine to the density of water]	Nominal
al	Albumin [presence of albumin in urine]	Nominal
su	Sugar [level of sugar present in urine]	Nominal
rbc	Red blood cells [red blood cells present in urine]	Nominal
pc	Pus cell [pus cells present in the urine, indicating major or minor infection]	Nominal
pcc	Pus cell clumps [pus cell clumps indicating if the infection is present in the urine]	Nominal
ba	Bacteria [if the growth of bacteria is evident in urine]	Nominal
bgr	Blood glucose random [sugar level in blood in mgs/dl]	Numerical
bu	Blood urea [level of urea nitrogen in blood in mgs/dl]	Numerical
sc	Serum creatinine [level of creatinine in blood in mgs/dl]	Numerical
sod	Sodium [level of sodium in blood in mEq/L]	Numerical
pot	Potassium [level of potassium in blood in mEq/L]	Numerical
hemo	Hemoglobin [protein in red blood cells in Gms]	Numerical
pcv	Packed cell volume [percentage of cells in blood]	Numerical
wc	White blood cell count [amount of white blood cells present in the blood (cells/cumm)]	Numerical
rc	Red blood cell count [amount of red blood cells present in the blood (millions/cmm)]	Nominal
htn	Hypertension [whether the patient has higher level of blood pressure]	Nominal
dm	Diabetes mellitus [presence of diabetes]	Nominal
cad	Coronary artery disease [whether the patient is suffering from coronary artery disease]	Nominal
appet	Appetite [loss of appetite]	Nominal
pe	Peda edema [level of leg swelling]	Nominal
ane	Anemia [whether the patient is suffering from anemia]	Nominal
Classification	Class [whether the patient has CKD or not]	Nominal

Table 3: Publicly available dataset used by researchers to predict CKD prognosis using AI

2. Data Preprocessing: The dataset was collected from a real. Figure 5 shows the number of non-null values present in the dataset.

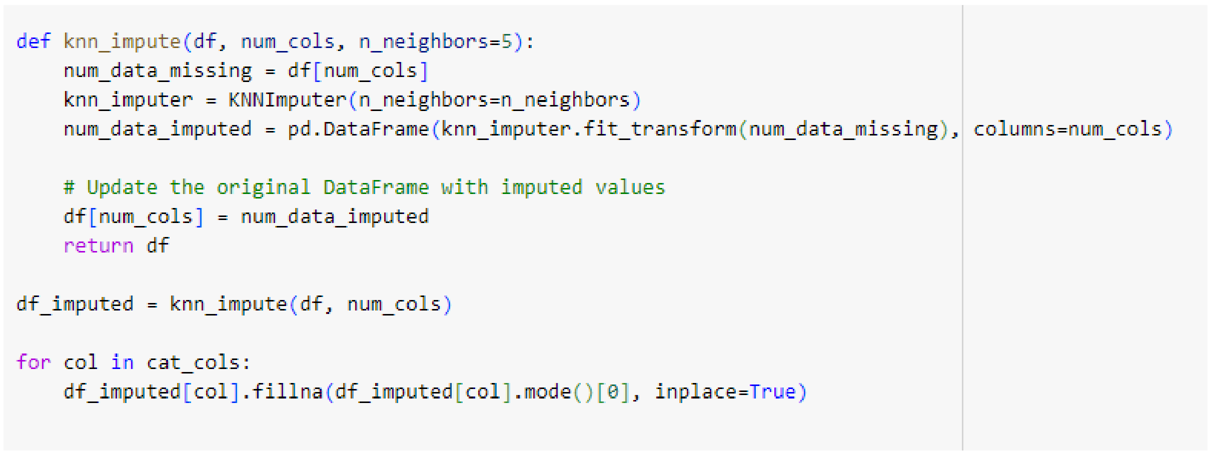


```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 400 entries, 0 to 399
Data columns (total 26 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   id              400 non-null    int64  
 1   age             391 non-null    float64
 2   bp    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

Figure 1: Number of non-null values in the dataset

It is observed that the dataset contains significant amount of missing readings, presence of outliers as well as class imbalance (250 belongs to CKD group whereas 150 belongs to non-CKD group). To handle this imbalance following methods were be used:

(a) Data Imputation: As the dataset is small (400 instances), data imputation approach is adopted. K-Nearest Neighbor imputation method was employed to handle missing numerical values as it replaces missing values by considering the 'k' nearest samples in the feature space. For the study, k was selected to be 5. Additionally, categorical data was filled using model imputation. The implementation for the numeric data was done using the 'KNNImputer' class from the the scikit-library in Python as described by the figure 6.



```python
def knn_impute(df, num_cols, n_neighbors=5):
    num_data_missing = df[num_cols]
    knn_imputer = KNNImputer(n_neighbors=n_neighbors)
    num_data_imputed = pd.DataFrame(knn_imputer.fit_transform(num_data_missing), columns=num_cols)

    # Update the original DataFrame with imputed values
    df[num_cols] = num_data_imputed
    return df

df_imputed = knn_impute(df, num_cols)

for col in cat_cols:
    df_imputed[col].fillna(df_imputed[col].mode()[0], inplace=True)
```

Figure 2: Data imputation techniques used in the study. KNN imputation for numeric data and mode imputation for categorical data

(b) Class imbalance: To solve the class imbalance in the dataset and avoid overfitting, a technique known as synthetic minority oversampling technique (SMOTE) was implemented. The technique was selected as it has shown satisfactory results in handling data imbalance by synthesizing new instance for the minority class using some neighboring samples from the minority class [40]. However, the SMOTE was only applied on the training dataset as it is mandatory that the evaluation is done on the real instances of the dataset. Thus, testing dataset was excluded from the SMOTE implementation.

(c) Label encoding: As can be seen from Tabel 6, there are many categorical features in the dataset. It is imperative to note that many of the features have been classified in the string format, for instance, 'ckd' or 'nockd'. As the machine learning algorithm employed in the study, such as random forest, require numeric values label encoder class from the scikit-library was implemented which converts categorical values into numeric values.

(d) Normalization: In the study, standard scaling was used to prevent features from larger scale values to dominate the the learning process. StandardScaler class of scikit-library was used after data imputation to maintain consistency and ensuring that all features contribute equally to the model [2].

3. Model Training and Evaluation: The study implements several supervised learning machine learning as described in section 5.2. Assuming the dataset is

pre-processed and ready to be used for training models, following describes the steps taken to train as well as metrics used to evaluate the results.

(a) 5-fold stratified cross-validation: Stratified cross-validation helps in model evaluation by splitting the dataset into several subsets of data known as folds. The split is created in such a way that each fold maintains the same class distribution [32]. In this study, 5-fold stratified cross-validation was employed to ensure that the evaluation metrics are reliable and avoid overfitting [32].

(b) SMOTE application: In each fold, SMOTE was implemented to only the training set to handle class imbalance by generating synthetic samples for the minority class.

(c) Scaling: The features in the training set were scaled using standard scaling and the same scaling parameters obtained from the training set were applied to the test set to ensure consistency.

(d) Model Training: The machine learning models were trained on the preprocessed training set and validated on the test set. The evaluation metrics from each fold were aggregated to compute the mean performance metrics for each model.

(e) Model Evaluation: Each model was evaluated on the validation set using the following metrics: accuracy, precision, recall, specificity, F1 score, and ROC-AUC. Refer to section 5.3 for more details.

(f) GridSearchCV: GridSearchCV is a hyperparameter optimization technique explores various pre-defined set of hyperparameters for a given algorithm and selects the best possible parameter values. In the study, it is used to mitigate the risk of overfitting by validating the model's performance on multiple folds of the training data [32].

(g) Feature Importance Calculation: For each model, feature importance was calculated based on the training process. 1) For tree-based Models (Decision Tree, Random Forest, XGBoost, LightGBM) feature importance was derived from the model's inherent feature importance attribute, which measures the contribution of each feature to the model's predictions.

    2) For Logistic Regression feature importance was determined by the absolute values of the model's coefficients, indicating the strength of the relationship between each feature and the target variable.

(h) Aggregation of Feature Importances: The final feature importance values from each fold were normalized and aggregated by computing the mean importance of each feature across all folds.

Figure 3 describes the steps taken in training and evaluation for the early prediction of CKD patients.

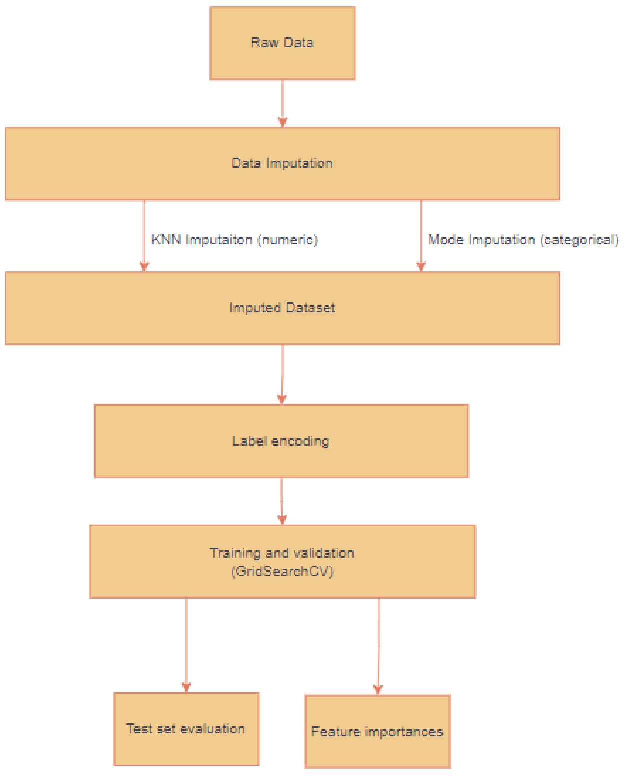


Figure 3: Methodology used for the UCI

(i) Model training and evaluation based on the best features: The study utilizes the top 6 features to train models described in section 5.2 and assess the performance of the models using a reduced feature set. This step was implemented to determine if a smaller subset of features can effectively predict CKD.

4. Code: To enable reproducibility of the outcomes, artefact code has been made available on GitHub. The link to the code is: https://github.com/souravsharm/CKD-detection-using-ML.git

5. Environment: For the development of the artefact, the study utilised Google Colab. This environment was chosen due to its cloud-based access, ease of use as well as sufficient computing capability. The programming language chosen was

Python due to its enormous library support for both machine learning as well as data manipulation oriented tasks. Specifically, scikit-learn, pandas, and numpy played crucial role in artefact development and evaluation.

3.5 Part 2 - Kidney Failure Prediction in CKD patients

3.5.1 Core Problem

Kidney failure is a severe stage of CKD that significantly impacts patients' health. Predicting kidney failure can help in managing the disease more effectively and preventing adverse outcomes. This part of the study aims to develop a machine learning model to predict kidney failure using clinical data from a different dataset than section 5.4.

3.5.2 Methodology

Data The study utilized longitudinal dataset that captures multiple values of estimated Glomerular Filtration Rate (eGFR) for each individual in the cohort. Table 4 and Tabel 5 describes main dataset and corresponding features respectively.

Dataset Name	Japanese-dataset
Cohort Size	1138
Dataset availability	Publicly available at: https://doi.org/10.5061/dryad.kq23s
Features	Age, sex, Systolic blood pressure, Body mass index, Haemoglobin, Albumin, serum creatinine (mg/dL), eGFR, dip-stick proteinuria, proteinuria, urinary occult blood, UPCR (urinary protein/creatinine ratio), UPCR category, hypertension, prevalence of CVD, diabetes, use of RAASi, use of CCB, use of diuretics, and HD or PD
Location	Japan
Number of kidney failure instances	Total: 162 (Male: 118, Female: 44)

Table 4: CKD dataset description for the Japanese dataset.

Attribute	Description	Units
age	Age in years	years
SBP	Diastolic blood pressure	mmHg
BMI	Body mass index	kg/m^2
Hb	Protein in red blood cells	g/dL
Alb	Presence of albumin in urine	g/dL
eGFR	Estimated Glomerular Filtration Rate at intervals of 6 months until 36 months	ml/min/1.73m^2
UPCR	Urinary Protein Creatinine Ratio	g/gCr

Table 5: Kidney failure (Japanese) dataset description with the number of missing values for each feature

The dataset contains significant amount of missing values and needs to be processed before training the model. The following section deals with the data-preprocessing steps undertaken during the study.

Data pre-processing

The initial datasets required data pre-processing to meet specific inclusion criteria. These criteria, defined in careful consultation with medical professionals as acknowledged earlier in the study, serve as filters to select only the most relevant features and outcomes, thereby enhancing the credibility and practical application of the resulting model.

In data pre-possessing, the following steps are followed: Three-variable Modification of Diet in Renal Disease equation was used in the calculation of eGFR, whereas only eGFR readings were recorded for successive visits except for the first measurement. Another problem was the inclusion of lot of empty fields. Also data was recorded in such a way that it would be difficult to directly use it for the model training/testing. Thus following steps were applied:

1. Column-Flattening: The original dataset had separate columns for eGFR measurements at different times ('eGFR(0M)', 'eGFR(6M)', etc.). These were flattened into a new DataFrame to create a uniform time-wise sequence of eGFR measurements for each patient.
2. Age-correction: Based on the time the eGFR measurements were taken, patients' ages were amended and rounded down to the nearest whole number.
3. Serum Creatinine Calculation: Since serum creatinine is essential for calculating eGFR using the CKD-EPI equation, its value was recalculated at each visit using the MDRD equation [41]. The following equation was employed for this recalculation.

   For males:
   $$SCr(mg/dL) = \left(\frac{eGFR}{194 \times \text{age}^{-0.287}}\right)^{-\frac{1}{1.094}} \tag{4}$$

   For females:
   $$SCr(mg/dL) = \left(\frac{eGFR}{194 \times \text{age}^{-0.287} \times 0.739}\right)^{-\frac{1}{1.094}} \tag{5}$$

In the CKD-EPI equation, serum creatinine is measured in micromoles per liter (µmol/L). To convert serum creatinine values from milligrams per deciliter (mg/dL) to µmol/L, the values are multiplied by a factor of 88.4. This conversion ensures that the units are consistent with those used in the CKD-EPI equation.

4. eGFR re-calculation using CKD-EPI equation: The eGFR was re-calculated using the CKD-EPI (Chronic Kidney Disease Epidemiology Collaboration) equation, which is a standard in nephrology. The following equations are used for the calculation [42]:

   For males:

$$eGFR = \begin{cases} 141 \times \left(\frac{SCr \times 0.0113}{0.9}\right)^{-0.411} \times 0.993^{\text{age}}, & \text{if } SCr \leq 80\,\mu\text{mol/L} \\ 141 \times \left(\frac{SCr \times 0.0113}{0.9}\right)^{-1.209} \times 0.993^{\text{age}}, & \text{if } SCr > 80\,\mu\text{mol/L} \end{cases} \tag{6}$$

   For females:

$$eGFR = \begin{cases} 144 \times \left(\frac{SCr \times 0.0113}{0.7}\right)^{-0.329} \times 0.993^{\text{age}}, & \text{if } SCr \leq 62\,\mu\text{mol/L} \\ 144 \times \left(\frac{SCr \times 0.0113}{0.7}\right)^{-1.209} \times 0.993^{\text{age}}, & \text{if } SCr > 62\,\mu\text{mol/L} \end{cases} \tag{7}$$

   The value of eGFR is in: mL/min/1.73m^2.

5. Implementing inclusion criteria: After implementing the above steps, inclusion criteria (as described in section 3.5.2) are applied. This results in the final dataset used for training/testing the model.

Assumptions for the Kidney failure prediction

1. Definition of Kidney Failure: To ensure accurate identification of kidney failure, this study adopts the criterion that an eGFR reading must be less than 15 mL/min/1.73m^2 on at least two separate occasions, each at least 30 days apart. This standard helps to exclude temporary fluctuations or anomalies in eGFR from being misclassified as kidney failure.

2. Measurement Methodology for eGFR: The CKD-EPI equation is employed to measure the estimated Glomerular Filtration Rate (eGFR), which is expressed in units of mL/min/1.73m^2. This equation is the most accepted and standardized methodology in the field of nephrology.

Inclusion criteria for the model training

1. Age Criteria: Only patients aged 18 years or over are included in the dataset.
2. eGFR Reading Criteria: A minimum of three eGFR readings is required for each patient's data to be included.
3. Initial eGFR Range: The initial eGFR measurement for each patient must fall within the range of 15 to 59 mL/min/1.73m^2.
4. Exclusion Criteria: Patients who have two eGFR readings below 15 mL/min/1.73m^2 on the same day, following their first eGFR reading, are excluded from the dataset.

Feature engineering Feature engineering plays a critical role in the predictive modeling of chronic kidney disease progression, especially when the focus of the research is on the effects of temporal variables. In this study, two specific engineered features are formulated: eGFR mean and eGFR slope as shown in Figure 4 depicts the calculation of eGFR slope and eGFR mean for Japanese-dataset.

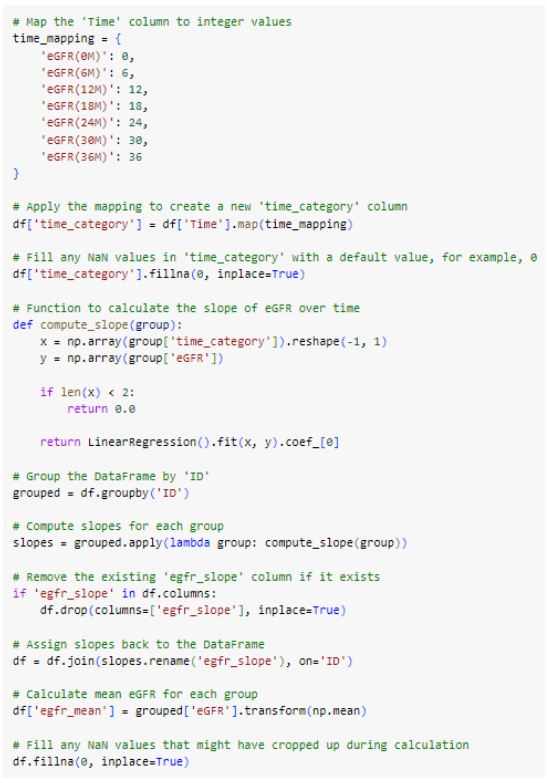
1. eGFR mean:

   The first engineered feature is the mean eGFR (egfr_mean) for each patient, calculated from their available eGFR readings. This feature aims to offer a generalized estimate of kidney function across multiple time points.

2. eGFR slope:

   Calculated using linear regression over time for each instance, the eGFR slope (egfr_slope) offers insights into the trend of a patient's kidney function-whether it is improving, deteriorating, or staying constant over time. Thus, the second feature is eGFR slope.

Both eGFR_mean and eGFR_slope bring into account the temporal dimension of eGFR readings, and thus are used to observe the impact on the model performance.



```
# Map the 'Time' column to integer values
time_mapping = {
    'eGFR(0M)': 0,
    'eGFR(6M)': 6,
    'eGFR(12M)': 12,
    'eGFR(18M)': 18,
    'eGFR(24M)': 24,
    'eGFR(30M)': 30,
    'eGFR(36M)': 36
}

# Apply the mapping to create a new 'time_category' column
df['time_category'] = df['Time'].map(time_mapping)

# Fill any NaN values in 'time_category' with a default value, for example, 0
df['time_category'].fillna(0, inplace=True)

# Function to calculate the slope of eGFR over time
def compute_slope(group):
    x = np.array(group['time_category']).reshape(-1, 1)
    y = np.array(group['eGFR'])

    if len(x) < 2:
        return 0.0

    return LinearRegression().fit(x, y).coef_[0]

# Group the DataFrame by 'ID'
grouped = df.groupby('ID')

# Compute slopes for each group
slopes = grouped.apply(lambda group: compute_slope(group))

# Remove the existing 'egfr_slope' column if it exists
if 'egfr_slope' in df.columns:
    df.drop(columns=['egfr_slope'], inplace=True)

# Assign slopes back to the DataFrame
df = df.join(slopes.rename('egfr_slope'), on='ID')

# Calculate mean eGFR for each group
df['egfr_mean'] = grouped['eGFR'].transform(np.mean)

# Fill any NaN values that might have cropped up during calculation
df.fillna(0, inplace=True)
```

Figure 4: eGFR slope and eGFR mean calculation.

Model training and evaluation   The model is trained using the models described in section 5.2. Similar to the 5.4, evaluation of 5-fold stratified cross-validation as well as GridSearchCV was done. Lastly, the final model performance was aggregated using the mean from each folds for a given machine learning model.

3.6 Performance evaluation

Depending on the situation, the prediction ability of the model varies greatly. For instance, model A might exhibit high sensitivity by correctly predicting patients that will have kidney failure whereas incorrectly predicts those who will not have such outcome. Therefore, the study utilises multiple criterion for performance evaluation. This provides more comprehensive insight into the performance of the model [39].

True positive (TP) refers to instances that are correctly classified as positive class by the model whereas true negative (TN) implies to the instances correctly classified as negative class. Similarly, false positive (FP) refers to instances that are incorrectly classified as positive class by the model whereas false negative (FN) implies to the instances incorrectly classified as negative class [43]. Keeping the above in mind, the study adopts six factors for evaluation:

1. Accuracy: This metrics is used to evaluate the overall performance of the model. Mathematically, it is the fraction of correct predictions among the total predictions made with value ranging from 0 to 1, with 1 representing that model predicts all positive and negative classes correctly, whereas a value of 0 suggests that model predicts all class instances incorrectly [43].

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{8}$$

2. Precision: Mathematically represented as the fraction of True Positives among the instances that the model predicted as positive. In other words, it quantifies the assessing ability of the model to correctly identify positive instance among all the instances it predicted as positive. Its value ranges from 0 to 1, with 1 representing that model predicts all the positive instance correctly whereas a value of 0 means that model predicts all the positive class instance incorrectly [43].

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{9}$$

3. Recall: Recall, also known as Sensitivity or True Positive Rate, evaluates model's ability to correctly identify positive instances among all the actual positive instances in the dataset. It is the ratio of correctly identified positive instances to the total number of actual positive instances in the dataset. The metric value ranges from 0 to 1, with 1 indicating that the model has perfectly identified all actual positive instances without any false negatives, while a value of 0 means that the model has failed to identify any of the actual positives [43].

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{10}$$

4. Specificity: Measure for the fraction of True Negatives among the actual negative instances is described by specificity. The value range of specificity is bound to [0,1], where higher values representing better ability to correctly identify negative class instances, whereas lower value indicating that the model has incorrectly classified all the actual negative instances as positive [43].

$$\text{Specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}} \tag{11}$$

5. F1 score: The harmonic mean of Precision and Recall is called F1 score. It is used with the aim to balance the two metrics (Precision and Recall). The value ranges from 0 to 1, with values closer to 1 indicating better model performance in terms of both false positives and false negatives [43].

$$\text{F1 score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} = \frac{2 \times \text{TP}}{2 \times \text{TP} + \text{FP} + \text{FN}} \tag{12}$$

6. ROC-AUC: It stands for Receiver Operating Characteristic-Area Under the Curve. This metric is used to quantify a model's ability to distinguish between the positive and negative classes. Essentially, the metric provides a single scalar value that represents the area under the ROC curve (a plot of the true positive rate against the false positive rate) across all possible thresholds. As this is a plot, no specific equation can be provided. The metric ranges from 0 to 1, where a value of 1 signifies model's ability to correctly classify all the positive instances from negative ones, and a value of 0.5 signifies no discriminatory power (akin to random guessing) [43].

Ethical considerations   Adoption of appropriate ethical considerations is a mandatory tasks that cannot be overstated. Especially when handling medical data, extra care and caution is considered due to sensitivity of data. In this study, appropriate ethical standards were maintained so that privacy and integrity are assured. Several steps were undertaken to withhold the ethic code when conducting research.

1. Data anonymity: The datasets utilized in the study exists is anonymous. No personal information that could be used to identify the person exists in the dataset.

2. Ethics committee approval: Japanese-dataset have ethics approval from all institutions that participated in the study and is registered in UMIN Clinical Trials Registry (UMIN000004461) [41].

3. Data storing: Data will be stored at Deakin University in Melbourne securely. Moreover, only secure and trusted channels such as Microsoft Teams are authorised used for even regular meetings with minimal dataset information.

4. Data usage: For both UCI-dataset and Japanese-dataset appropriate usage principles were followed as ut allows re-use and transformation of data specified

under Creative Commons Attribution License available online at: https://creativecommons.org/licenses/by/4.0/.

3.7 Conclusion

With the implementation of this methodology, the study aims to provide comprehensive insights into the impact of temporal data on the effectiveness of predictive models for kidney failure.

4 Artefact Development Approach

This chapter describes the different artefacts created in the study. Artefacts are the code for data pre-processing as well as model development of decision tree, random forest, extreme gradient boosting (XGBoost), and light gradient boosting machine (LGBM) along with respective hyper-parameters used for tuning.

In conclusion, the model employs a stratified k-fold cross-validation method and the one with the best hyper-parameters is trained on the entire dataset for deployment or future use.

5 Empirical Evaluation

This chapter of the study critically evaluate the machine learning models described in the previous chapter. Most of the current studies evaluate model based on accuracy, recall, precision, F1 score, specificity and ROC-AUC, as the metrics provide more comprehensive performance insight of the model. [39][39][25][29].

5.1 Introduction

Chronic kidney disease (CKD) is a significant global health issue due to its prevalence worldwide. The fact that CKD does not have a cure and that the detailed checkup require blood and urine samples for determining its presence/progression makes CKD a very challenging subject for researchers [31]. This study evaluates the effectiveness of machine learning models developed for two primary objectives: early detection of CKD (referred to as UCI model as it used UCI dataset) and prediction of kidney failure (referred to as Japanese model as it used Japanese dataset). The aim of this section deals is to assess the performance and reliability of the model predictions.

5.2 Model Implementation

The following models are used in the study:

1. Decision Tree (DT)
2. Random Forest (RF)
3. Logistic Regression (LR)
4. Extreme gradient boosting machine (XGBoost)
5. Light gradient boosting machine (LightGB)

5.3 Performance evaluation

Depending on the situation, the prediction ability of the model varies greatly. For instance, model A might exhibit high sensitivity by correctly predicting patients that will have kidney failure whereas incorrectly predicts those who will not have such outcome. Therefore, the study utilises multiple criterion for performance evaluation. This provides more comprehensive insight into the performance of the model [39].

True positive (TP) refers to instances that are correctly classified as positive class by the model whereas true negative (TN) implies to the instances correctly classified as negative class. Similarly, false positive (FP) refers to instances that are incorrectly classified as positive class by the model whereas false negative (FN) implies to the instances incorrectly classified as negative class [43]. Keeping the above in mind, the study adopts six factors for evaluation:

1. Accuracy: This metrics is used to evaluate the overall performance of the model. Mathematically, it is the fraction of correct predictions among the total predictions made with value ranging from 0 to 1, with 1 representing that model

predicts all positive and negative classes correctly, whereas a value of 0 suggests that model predicts all class instances incorrectly [43].

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{13}$$

2. Precision: Mathematically represented as the fraction of True Positives among the instances that the model predicted as positive. In other words, it quantifies the assessing ability of the model to correctly identify positive instance among all the instances it predicted as positive. Its value ranges from 0 to 1, with 1 representing that model predicts all the positive instance correctly whereas a value of 0 means that model predicts all the positive class instance incorrectly [43].

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{14}$$

3. Recall: Recall, also known as Sensitivity or True Positive Rate, evaluates model's ability to correctly identify positive instances among all the actual positive instances in the dataset. It is the ratio of correctly identified positive instances to the total number of actual positive instances in the dataset. The metric value ranges from 0 to 1, with 1 indicating that the model has perfectly identified all actual positive instances without any false negatives, while a value of 0 means that the model has failed to identify any of the actual positives [43].

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{15}$$

4. Specificity: Measure for the fraction of True Negatives among the actual negative instances is described by specificity. The value range of specificity is bound to [0,1], where higher values representing better ability to correctly identify negative class instances, whereas lower value indicating that the model has incorrectly classified all the actual negative instances as positive [43].

$$\text{Specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}} \tag{16}$$

5. F1 score: The harmonic mean of Precision and Recall is called F1 score. It is used with the aim to balance the two metrics (Precision and Recall). The value ranges from 0 to 1, with values closer to 1 indicating better model performance in

terms of both false positives and false negatives [43].

$$\text{F1 score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} = \frac{2 \times \text{TP}}{2 \times \text{TP} + \text{FP} + \text{FN}} \tag{17}$$

6. ROC-AUC: It stands for Receiver Operating Characteristic-Area Under the Curve. This metric is used to quantify a model's ability to distinguish between the positive and negative classes. Essentially, the metric provides a single scalar value that represents the area under the ROC curve (a plot of the true positive rate against the false positive rate) across all possible thresholds. As this is a plot, no specific equation can be provided. The metric ranges from 0 to 1, where a value of 1 signifies model's ability to correctly classify all the positive instances from negative ones, and a value of 0.5 signifies no discriminatory power (akin to random guessing) [43].

5.4 Part 1 - Early Detection of CKD

5.4.1 Core Problem

Early detection is critical for timely intervention and management of the disease [39]. By using the clinical dataset, the study aims to develop several machine learning models to predict the whether the patient is likely to develop chronic kidney disease or not.

5.4.2 Environment and code availability

1. Code: To enable reproducibility of the outcomes, artefact code has been made available on GitHub. The link to the code is: https://github.com/souravsharm/CKD-detection-using-ML.git
2. Environment: For the development of the artefact, the study utilised Google Colab. This environment was chosen due to its cloud-based access, ease of use as well as sufficient computing capability. The programming language chosen was Python due to its enormous library support for both machine learning as well as

data manipulation oriented tasks. Specifically, scikit-learn, pandas, and numpy played crucial role in artefact development and evaluation.

5.4.3 Data and Results

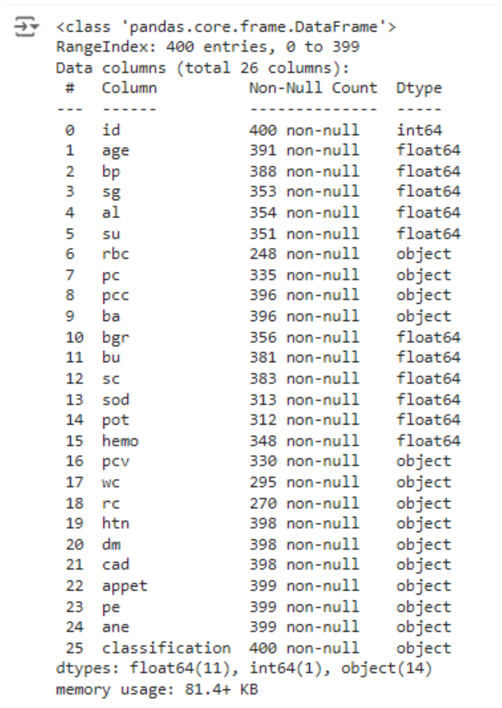
There are main requirements for replicating the study outcomes as described below:

1. Data: The dataset for the study is obtained from University of California Irvine (UCI) Machine Learning Repository available online at https://tinyurl.com/UCI-dataset-public. The dataset contained clinical readings of 400 patients collected from the Apollo Hospital, Karaikudi, India during a nearly 2-month period in 2015. With total number of 25 attributes in total with 11 numeric, 13 nominal features, and 1 target feature (notCKD/CKD) as described in table 6.

Attribute	Name	Category
age	Age [age in years]	Numerical
bp	Blood pressure [diastolic blood pressure in mm/Hg]	Numerical
sg	Specific gravity [specific gravity to compare the density of urine to the density of water]	Nominal
al	Albumin [presence of albumin in urine]	Nominal
su	Sugar [level of sugar present in urine]	Nominal
rbc	Red blood cells [red blood cells present in urine]	Nominal
pc	Pus cell [pus cells present in the urine, indicating major or minor infection]	Nominal
pcc	Pus cell clumps [pus cell clumps indicating if the infection is present in the urine]	Nominal
ba	Bacteria [if the growth of bacteria is evident in urine]	Nominal
bgr	Blood glucose random [sugar level in blood in mgs/dl]	Numerical
bu	Blood urea [level of urea nitrogen in blood in mgs/dl]	Numerical
sc	Serum creatinine [level of creatinine in blood in mgs/dl]	Numerical
sod	Sodium [level of sodium in blood in mEq/L]	Numerical
pot	Potassium [level of potassium in blood in mEq/L]	Numerical
hemo	Hemoglobin [protein in red blood cells in Gms]	Numerical
pcv	Packed cell volume [percentage of cells in blood]	Numerical
wc	White blood cell count [amount of white blood cells present in the blood (cells/cumm)]	Numerical
rc	Red blood cell count [amount of red blood cells present in the blood (millions/cmm)]	Nominal
htn	Hypertension [whether the patient has higher level of blood pressure]	Nominal
dm	Diabetes mellitus [presence of diabetes]	Nominal
cad	Coronary artery disease [whether the patient is suffering from coronary artery disease]	Nominal
appet	Appetite [loss of appetite]	Nominal
pe	Peda edema [level of leg swelling]	Nominal
ane	Anemia [whether the patient is suffering from anemia]	Nominal
Classification	Class [whether the patient has CKD or not]	Nominal

Table 6: Publicly available dataset used by researchers to predict CKD prognosis using AI

2. Data Preprocessing: The dataset was collected from a real. Figure 5 shows the number of non-null values present in the dataset.

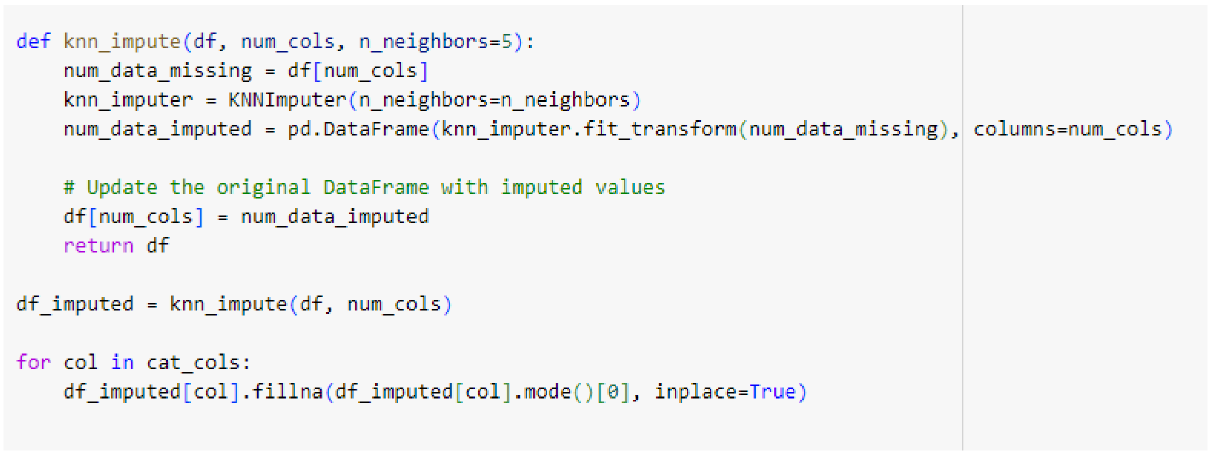


```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 400 entries, 0 to 399
Data columns (total 26 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   id              400 non-null    int64  
 1   age             391 non-null    float64
 2   bp    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

Figure 5: Number of non-null values in the dataset

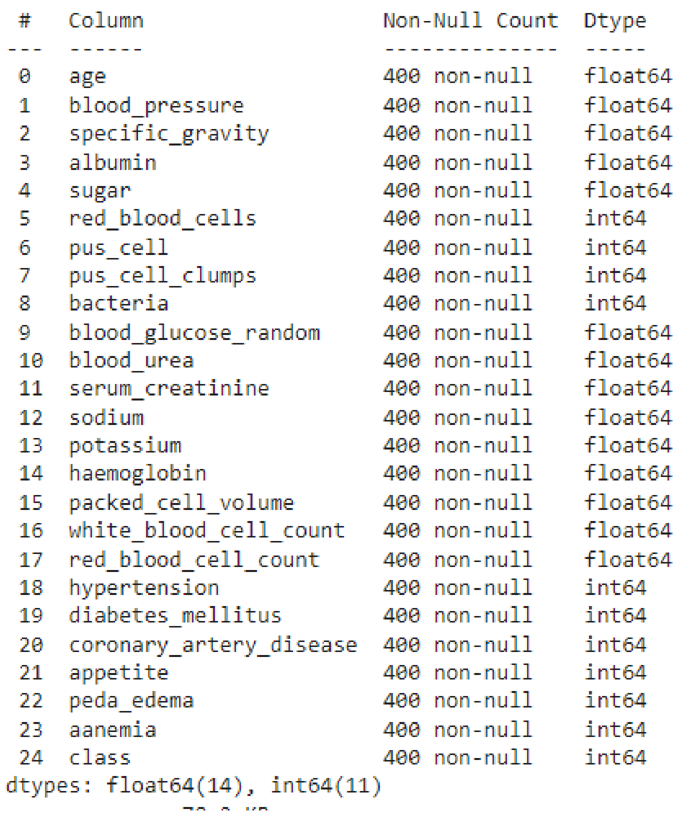
It is observed that the dataset contains significant amount of missing readings, presence of outliers as well as class imbalance (250 belongs to CKD group whereas 150 belongs to non-CKD group). To handle this imbalance following methods were be used:

(a) Data Imputation: As the dataset is small (400 instances), data imputation approach is adopted. K-Nearest Neighbor imputation method was employed to handle missing numerical values as it replaces missing values by considering the 'k' nearest samples in the feature space. For the study, k was selected to be 5. Additionally, categorical data was filled using model imputation. The implementation for the numeric data was done using the 'KNNImputer' class from the the scikit-library in Python as described by the figure 6. The result of the data imputation technique is shown in the figure 7



```python
def knn_impute(df, num_cols, n_neighbors=5):
    num_data_missing = df[num_cols]
    knn_imputer = KNNImputer(n_neighbors=n_neighbors)
    num_data_imputed = pd.DataFrame(knn_imputer.fit_transform(num_data_missing), columns=num_cols)

    # Update the original DataFrame with imputed values
    df[num_cols] = num_data_imputed
    return df

df_imputed = knn_impute(df, num_cols)

for col in cat_cols:
    df_imputed[col].fillna(df_imputed[col].mode()[0], inplace=True)
```

Figure 6: Code block for KNN imputation for numeric data and mode imputation for categorical data

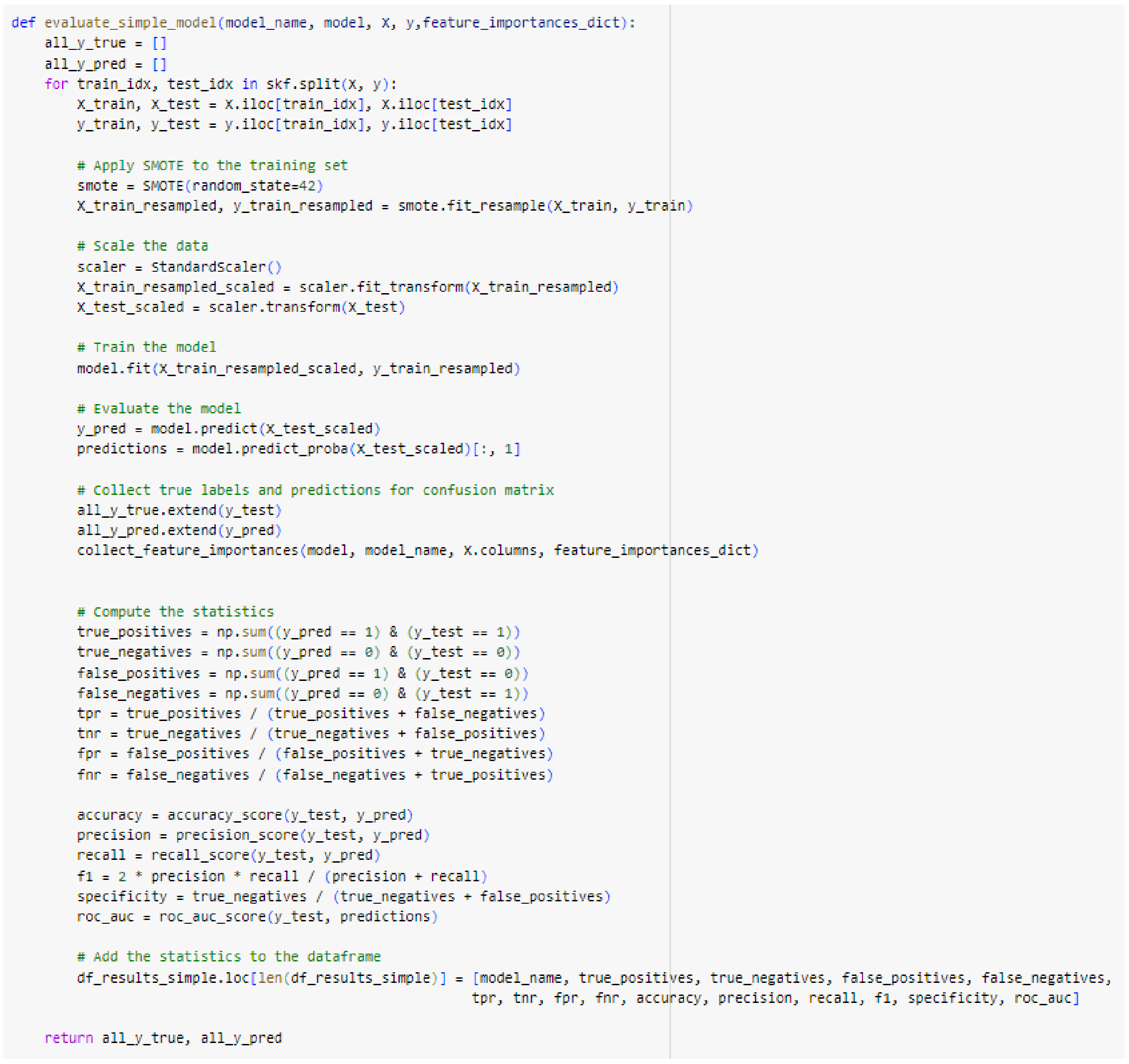


```
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   age                      400 non-null    float64
 1   blood_pressure           400 non-null    float64
 2   specific_gravity         400 non-null    float64
 3   albumin                  400 non-null    float64
 4   sugar                    400 non-null    float64
 5   red_blood_cells          400 non-null    int64
 6   pus_cell                 400 non-null    int64
 7   pus_cell_clumps          400 non-null    int64
 8   bacteria                 400 non-null    int64
 9   blood_glucose_random     400 non-null    float64
 10  blood_urea               400 non-null    float64
 11  serum_creatinine         400 non-null    float64
 12  sodium                   400 non-null    float64
 13  potassium                400 non-null    float64
 14  haemoglobin              400 non-null    float64
 15  packed_cell_volume       400 non-null    float64
 16  white_blood_cell_count   400 non-null    float64
 17  red_blood_cell_count     400 non-null    float64
 18  hypertension             400 non-null    int64
 19  diabetes_mellitus        400 non-null    int64
 20  coronary_artery_disease  400 non-null    int64
 21  appetite                 400 non-null    int64
 22  peda_edema               400 non-null    int64
 23  aanemia                  400 non-null    int64
 24  class                    400 non-null    int64
dtypes: float64(14), int64(11)
```

Figure 7: Dataset after imputation techniques used in the study. KNN imputation for numeric data and mode imputation for categorical data

(b) Class imbalance: To solve the class imbalance in the dataset and avoid overfitting, a technique known as synthetic minority oversampling technique

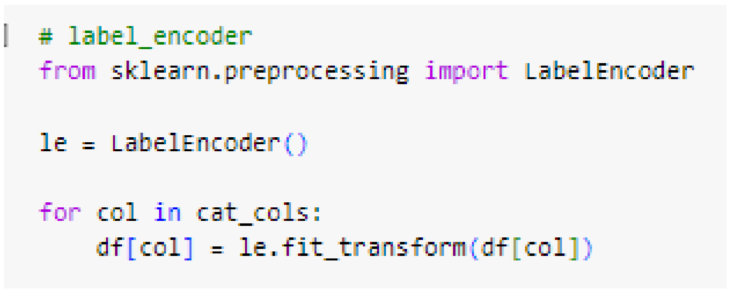
(SMOTE) was implemented. The technique was selected as it has shown satisfactory results in handling data imbalance by synthesizing new instance for the minority class using some neighboring samples from the minority class [40]. However, the SMOTE was only applied on the training dataset as it is mandatory that the evaluation is done on the real instances of the dataset. Thus, testing dataset was excluded from the SMOTE implementation. The code implementation for the same is describe in figure 8



```python
def evaluate_simple_model(model_name, model, X, y,feature_importances_dict):
    all_y_true = []
    all_y_pred = []
    for train_idx, test_idx in skf.split(X, y):
        X_train, X_test = X.iloc[train_idx], X.iloc[test_idx]
        y_train, y_test = y.iloc[train_idx], y.iloc[test_idx]

        # Apply SMOTE to the training set
        smote = SMOTE(random_state=42)
        X_train_resampled, y_train_resampled = smote.fit_resample(X_train, y_train)

        # Scale the data
        scaler = StandardScaler()
        X_train_resampled_scaled = scaler.fit_transform(X_train_resampled)
        X_test_scaled = scaler.transform(X_test)

        # Train the model
        model.fit(X_train_resampled_scaled, y_train_resampled)

        # Evaluate the model
        y_pred = model.predict(X_test_scaled)
        predictions = model.predict_proba(X_test_scaled)[:, 1]

        # Collect true labels and predictions for confusion matrix
        all_y_true.extend(y_test)
        all_y_pred.extend(y_pred)
        collect_feature_importances(model, model_name, X.columns, feature_importances_dict)


        # Compute the statistics
        true_positives = np.sum((y_pred == 1) & (y_test == 1))
        true_negatives = np.sum((y_pred == 0) & (y_test == 0))
        false_positives = np.sum((y_pred == 1) & (y_test == 0))
        false_negatives = np.sum((y_pred == 0) & (y_test == 1))
        tpr = true_positives / (true_positives + false_negatives)
        tnr = true_negatives / (true_negatives + false_positives)
        fpr = false_positives / (false_positives + true_negatives)
        fnr = false_negatives / (false_negatives + true_positives)

        accuracy = accuracy_score(y_test, y_pred)
        precision = precision_score(y_test, y_pred)
        recall = recall_score(y_test, y_pred)
        f1 = 2 * precision * recall / (precision + recall)
        specificity = true_negatives / (true_negatives + false_positives)
        roc_auc = roc_auc_score(y_test, predictions)

        # Add the statistics to the dataframe
        df_results_simple.loc[len(df_results_simple)] = [model_name, true_positives, true_negatives, false_positives, false_negatives,
                                                         tpr, tnr, fpr, fnr, accuracy, precision, recall, f1, specificity, roc_auc]

    return all_y_true, all_y_pred
```

Figure 8: Data imputation techniques used in the study. KNN imputation for numeric data and mode imputation for categorical data

(c) Label encoding: As can be seen from Tabel 6, there are many categorical features in the dataset. It is imperative to note that many of the features have been classified in the string format, for instance, 'ckd' or 'nockd'. As the machine learning algorithm employed in the study, such as random forest, require numeric values label encoder class from the scikit-library was implemented which converts categorical values into numeric values. Figure 9 describes the code block for the implementation of Label encoder from the scikit-library.



```
# label_encoder
from sklearn.preprocessing import LabelEncoder

le = LabelEncoder()

for col in cat_cols:
    df[col] = le.fit_transform(df[col])
```

Figure 9: Data imputation techniques used in the study. KNN imputation for numeric data and mode imputation for categorical data

(d) Normalization: In the study, standard scaling was used to prevent features from larger scale values to dominate the the learning process. StandardScaler class of scikit-library was used after data imputation to maintain consistency and ensuring that all features contribute equally to the model [2]. Implementation for the StandardScaler is illustrated in fig 8

3. Model Training and Evaluation: The study implements several supervised learning machine learning as described in section 5.2. Assuming the dataset is pre-processed and ready to be used for training models, following describes the steps taken to train as well as metrics used to evaluate the results.

(a) 5-fold stratified cross-validation: Stratified cross-validation helps in model evaluation by splitting the dataset into several subsets of data known as folds. The split is created in such a way that each fold maintains the same class distribution [32]. In this study, 5-fold stratified cross-validation was employed to ensure that the evaluation metrics are reliable and avoid overfitting [32].

(b) SMOTE application: In each fold, SMOTE was implemented to only the training set to handle class imbalance by generating synthetic samples for the minority class.

(c) Scaling: The features in the training set were scaled using standard scaling and the same scaling parameters obtained from the training set were applied to the test set to ensure consistency.

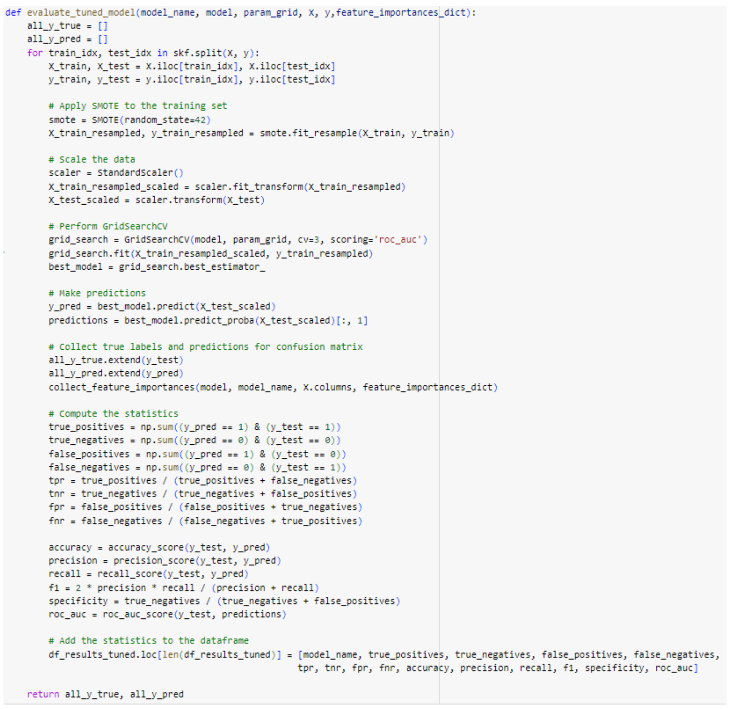
(d) Model Training: The machine learning models were trained on the preprocessed training set and validated on the test set. The evaluation metrics from each fold were aggregated to compute the mean performance metrics for each model.

(e) Model Evaluation: Each model was evaluated on the validation set using the following metrics: accuracy, precision, recall, specificity, F1 score, and ROC-AUC. Refer to section 5.3 for more details. Table 7 demonstrate the model performance based on the metric.

Model	Accuracy	Precision	Recall	f1_score	Specificity	ROC-AUC
DecisionTree	0.950	0.917462	0.953333	0.934617	0.948	0.950667
LightGBM	0.965	0.937097	0.973333	0.954733	0.960	0.995467
LogisticRegression	0.980	0.968347	0.980000	0.973643	0.980	0.998800
RandomForest	0.975	0.979048	0.953333	0.965747	0.988	0.998600
XGBoost	0.970	0.966407	0.953333	0.959655	0.980	0.996933

Table 7: Performance metrics of Simple models in predicting CKD class in UCI dataset

(f) GridSearchCV: GridSearchCV is a hyperparameter optimization technique explores various pre-defined set of hyperparameters for a given algorithm and selects the best possible parameter values. In the study, it is used to mitigate the risk of overfitting by validating the model's performance on multiple folds of the training data [32]. Figure 10 illustrates the code implementation of hyperparameter tuned model.



```python
def evaluate_tuned_model(model_name, model, param_grid, X, y,feature_importances_dict):
    all_y_true = []
    all_y_pred = []
    for train_idx, test_idx in skf.split(X, y):
        X_train, X_test = X.iloc[train_idx], X.iloc[test_idx]
        y_train, y_test = y.iloc[train_idx], y.iloc[test_idx]

        # Apply SMOTE to the training set
        smote = SMOTE(random_state=42)
        X_train_resampled, y_train_resampled = smote.fit_resample(X_train, y_train)

        # Scale the data
        scaler = StandardScaler()
        X_train_resampled_scaled = scaler.fit_transform(X_train_resampled)
        X_test_scaled = scaler.transform(X_test)

        # Perform GridSearchCV
        grid_search = GridSearchCV(model, param_grid, cv=3, scoring='roc_auc')
        grid_search.fit(X_train_resampled_scaled, y_train_resampled)
        best_model = grid_search.best_estimator_

        # Make predictions
        y_pred = best_model.predict(X_test_scaled)
        predictions = best_model.predict_proba(X_test_scaled)[:, 1]

        # Collect true labels and predictions for confusion matrix
        all_y_true.extend(y_test)
        all_y_pred.extend(y_pred)
        collect_feature_importances(model, model_name, X.columns, feature_importances_dict)

        # Compute the statistics
        true_positives = np.sum((y_pred == 1) & (y_test == 1))
        true_negatives = np.sum((y_pred == 0) & (y_test == 0))
        false_positives = np.sum((y_pred == 1) & (y_test == 0))
        false_negatives = np.sum((y_pred == 0) & (y_test == 1))
        tpr = true_positives / (true_positives + false_negatives)
        tnr = true_negatives / (true_negatives + false_positives)
        fpr = false_positives / (false_positives + true_negatives)
        fnr = false_negatives / (false_negatives + true_positives)

        accuracy = accuracy_score(y_test, y_pred)
        precision = precision_score(y_test, y_pred)
        recall = recall_score(y_test, y_pred)
        f1 = 2 * precision * recall / (precision + recall)
        specificity = true_negatives / (true_negatives + false_positives)
        roc_auc = roc_auc_score(y_test, predictions)

        # Add the statistics to the dataframe
        df_results_tuned.loc[len(df_results_tuned)] = [model_name, true_positives, true_negatives, false_positives, false_negatives,
                                                        tpr, tnr, fpr, fnr, accuracy, precision, recall, f1, specificity, roc_auc]

    return all_y_true, all_y_pred
```

Figure 10: Codeblock for the GridSearchCV for the UCI dataset

Model	Version	Accuracy	Precision	Recall	f1_score	Specificity	ROC-AUC
DecisionTree	Simple	0.950	0.917462	0.953333	0.934617	0.948	0.950667
	Tuned	0.945	0.910659	0.946667	0.928179	0.944	0.944533
LightGBM	Simple	0.965	0.937097	0.973333	0.954733	0.960	0.995467
	Tuned	0.970	0.947742	0.973333	0.960328	0.968	0.996667
Logistic Regression	Simple	0.980	0.968347	0.980000	0.973643	0.980	0.998800
	Tuned	0.970	0.926837	1.000000	0.961794	0.952	0.999733
RandomForest	Simple	0.975	0.979048	0.953333	0.965747	0.988	0.998600
	Tuned	0.975	0.979048	0.953333	0.965747	0.988	0.998600
XGBoost	Simple	0.970	0.966407	0.953333	0.959655	0.980	0.996933
	Tuned	0.975	0.973319	0.960000	0.966322	0.984	0.996733

Table 8: Performance of tuned model by using GridSearchCV vs Siimple models with default classifiers

Tabel 8 compares the performance of tuned model by using GridSearchCV and simple models with default classifiers.

(g) Feature Importance Calculation: For each model, feature importance was calculated based on the training process. 1) For tree-based Models (Decision Tree, Random Forest, XGBoost, LightGBM) feature importance was derived from the model's inherent feature importance attribute, which measures the contribution of each feature to the model's predictions.

2) For Logistic Regression feature importance was determined by the absolute values of the model's coefficients, indicating the strength of the relationship between each feature and the target variable.

(h) Aggregation of Feature Importances: The final feature importance values from each fold were normalized and aggregated by computing the mean importance of each feature across all folds. Figure 11 and Tabel 9illustrates the aggregated feature importance of the machine learning algorithms used in early predicting of CKD.

Feature	Importance
Haemoglobin	0.282
Specific gravity	0.145
Serum creatinine	0.090
Albumin	0.071
Packed cell volume	0.069
Diabetes mellitus	0.048
Red blood cell count	0.040

Table 9: Overall Feature Importances (from highest to lowest) early predicting of CKD

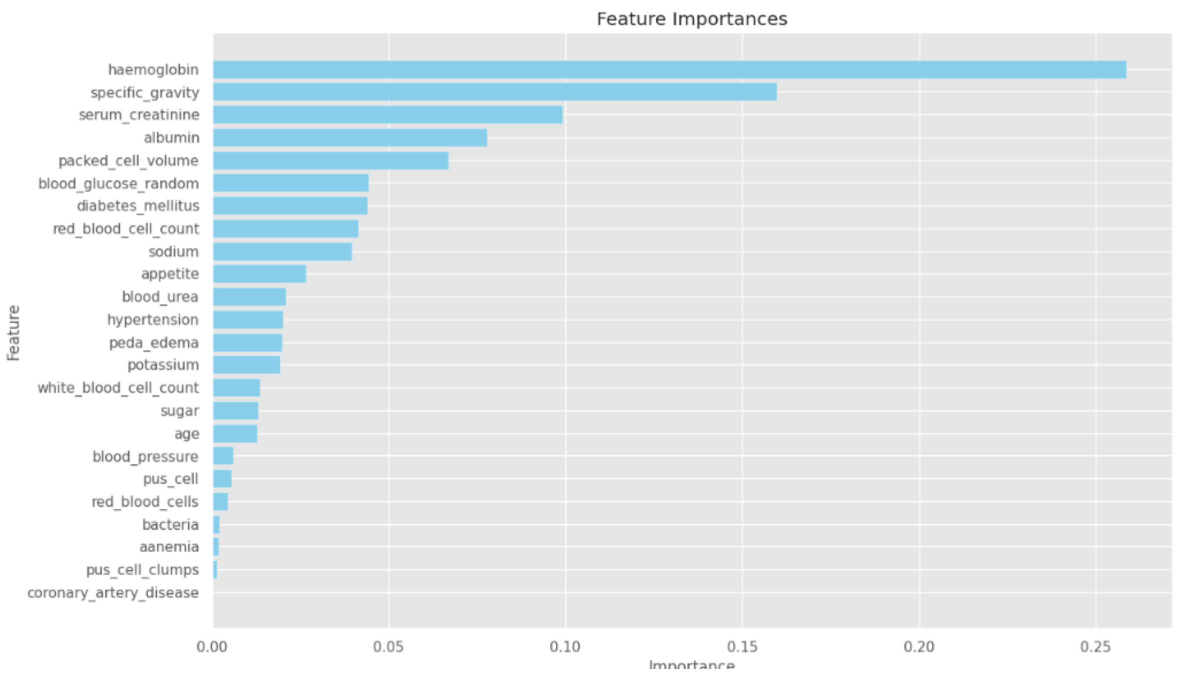


Figure 11: Plot of the best to least important features to predict CKD

(i) Model training and evaluation based on the best features: The study utilizes the top 6 features to train models described in section 5.2 and assess the performance of the models using a reduced feature set. This step was implemented to determine if a smaller subset of features can effectively predict CKD. Tabel 10 demonstrate the performance metrics of different models using only the top 7 features.

Model	Accuracy	Precision	Recall	F1 Score	Specificity	ROC-AUC
Logistic Regression	0.9675	0.9493	0.9667	0.9570	0.9680	0.9955
Decision Tree	0.9500	0.9308	0.9400	0.9334	0.9560	0.9480
XGBoost	0.9675	0.9624	0.9533	0.9568	0.9760	0.9987
LightGBM	0.9825	0.9867	0.9667	0.9763	0.9920	0.9993
Random Forest	0.9775	0.9795	0.9600	0.9691	0.9880	0.9989

Table 10: Performance metrics of models with the top 7 features

6 Results & Discussion

In the comparison of kidney failure prediction models, the best model is the simple XGBoost model, which achieved the highest performance metrics across the board. The simple XGBoost model demonstrated an accuracy of 0.9950, precision of 0.9870, recall of 0.9758, F1 score of 0.9812, specificity of 0.9979, and ROC-AUC of 0.9977. These metrics indicate that the simple XGBoost model performs exceptionally well in distinguishing between kidney failure and non-kidney failure cases.

Study	Aim[1]	Key Metrics[2]
Wang et al [30]	End-stage renal disease (ESRD) in patients with type 2 diabetes mellitus (T2DM)	XGBoost- AUROC: 95%
Wainstein et al [44]	Risk of ESKD was calculated using the 4-variable KFRE	Four Random Forest- AUROC: 97–98%
Ferguson et al [29]	40% decline in eGFR or kidney failure	Random Forest- AUCROC: 88%
This study	Kidney failure[3] prediction using temporal data	LightGBM- AUROC: 100% (Tuned) XGBoost- AUROC: 99.77% (Tuned)

Table 11: Comparison of Studies

[1]ESRD/kidney failure: eGFR < 15 ml/min/1.73m^2

[2]Only those metrics are used that is available for comparison

[3]Kidney failure: eGFR < 15 ml/min/1.73m^2 for two separate occasions 30 days apart

6.1 Novelty of the Study

The key innovative feature of this research lies in the utilization of AI/ML techniques to understand the role of temporal data in predicting kidney failure. Unlike existing studies, this research not only deploys state-of-the-art machine learning algorithms like LightGBM but also engages in a comparative analysis with the current top findings in the field.This comprehensive approach addresses the research question and fills a significant gap in existing literature by offering both a novel focus and a robust methodological framework.

6.2 Comparative Analysis and Validation

While direct comparison with other studies is limited due to the unique aspects of this research, the findings do indicate that the implemented models outperform existing state-of-the-art algorithms, particularly in terms of the AUROC metric.

6.3 Limitations

Even though the study developed provides significant contributions to the field of AI/ML-based kidney failure prediction by examining the impact of temporal data, it is important to acknowledge the underlying limitations. The following limitations of the study:

1. Lack of other variables: The study does not take into account the other variables which may provide useful information such as comorbidity-associations over time.
2. Hyper-parameter tuning: Although the study utilized hyper-parameter tuning, more research is needed on other hyper-parameters as well as optimization techniques.
3. Dataset: Both datasets are small and thus even after employing optimization strategies to reduce overfitting, it is possible that the model is still overfitting. Thus the external validation is required.

These limitations server as a reference points for future works and are required to be addressed to build a more comprehensive generalized tool for kidney failure prediction.

7 Conclusion & Future Work

In conclusion, the study found the following observations:

1. For CKD detection, the best-performing model was LightGBM.This model demonstrated exceptional performance, maintaining high accuracy, precision, recall, F1 score, specificity, and ROC-AUC values across both the full feature set and the reduced feature set. Specifically, in the context of using the top 7 features, LightGBM achieved an accuracy of 98%, precision of 99%, recall of 97%, F1 score of 98%, specificity of 99%, and ROC-AUC of 100%. This consistency indicates LightGBM's robustness and efficiency in handling critical features effectively, making it the optimal choice for CKD detection.

2. For kidney failure prediction, the best-performing model was XGBoost, both in its simple and tuned versions. The simple XGBoost model achieved an accuracy of 99.50%, F1 score of 98.12%, and specificity of 99.79%. The tuned XGBoost model showed similarly high performance, with an accuracy of 99.47%, F1 score of 98.00%, and specificity of 99.79%. These results indicate that it is suitable for kidney failure prediction model.

3. From the above, it can be concluded that egfr_mean and egfr_slope can be helpful in predicting kidney failure for CKD patients.

4. The study also discovered features listed in Table 5 can be used to reliably predict CKD instance.

7.1 Future works

The limitations provide not only a reflection but also a guiding path for the future works. The future works include several important aspects such as interpretability, training on more diverse dataset, incorporating other hyper-parameters and optimization techniques, use of transfer learning to make more generalized model for kidney failure prediction and observe feature importance of each variable to build better prediction tools.

1. Interpretability: Further exploration of the models' decision-making mechanisms using techniques such as SHAP and LIME.

2. Diverse Datasets: Expand the training sets to include a broader range of demographic and clinical variables to test the models' robustness and applicability.

3. Hyper-Parameters and Optimization Techniques: Investigate the impact of different hyper-parameters and employ various optimization techniques to improve model performance.

4. Transfer Learning: Utilize pre-trained models and fine-tune them for the specific task of kidney failure prediction, aiming for a more generalizable solution.

5. Feature Importance: Examine the contribution of each variable to the predictive model, aiming to refine and potentially simplify the models for future use.

Employing and incorporating these points will not only fill the research gap but will form the foundation for the future works.

References

[1] M. S. Arif, A. Mukheimer, and D. Asif, "Enhancing the early detection of chronic kidney disease: A robust machine learning model," Big Data and Cognitive Computing, vol. 7, no. 3, 2023. [Online]. Available: https://www.mdpi.com/2504-2289/7/3/144

[2] R. Sawhney, A. Malik, S. Sharma, and V. Narayan, "A comparative assessment of artificial intelligence models used for early prediction and evaluation of chronic kidney disease," Decision Analytics Journal, vol. 6, p. 100169, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2772662223000097

[3] A. A. Honeycutt, J. E. Segel, X. Zhuo, T. J. Hoerger, K. Imai, and D. Williams, "Medical costs of ckd in the medicare population," Journal of the American Society of Nephrology: JASN, vol. 24, no. 9, p. 1478, 2013.

[4] A. S. Levey and J. Coresh, "Chronic kidney disease," The lancet, vol. 379, no. 9811, pp. 165–180, 2012.

[5] D. Swain, U. Mehta, A. Bhatt, H. Patel, K. Patel, D. Mehta, B. Acharya, V. C. Gerogiannis, A. Kanavos, and S. Manika, "A robust chronic kidney disease classifier using machine learning," Electronics, vol. 12, no. 1, p. 212, 2023.

[6] R. Sawhney, A. Malik, S. Sharma, and V. Narayan, "A comparative assessment of artificial intelligence models used for early prediction and evaluation of chronic kidney disease. dec. anal. j. 6, 100169 (2023)," 2023.

[7] M. Almasoud and T. E. Ward, "Detection of chronic kidney disease using machine learning algorithms with least number of predictors," International Journal of Soft Computing and Its Applications, vol. 10, no. 8, 2019.

[8] A. J. Aljaaf, D. Al-Jumeily, H. M. Haglan, M. Alloghani, T. Baker, A. J. Hussain, and J. Mustafina, "Early prediction of chronic kidney disease using machine learning supported by predictive analytics," in 2018 IEEE congress on evolutionary computation (CEC). IEEE, 2018, pp. 1–9.

[9] M. Rashed-Al-Mahfuz, A. Haque, A. Azad, S. A. Alyami, J. M. Quinn, and M. A. Moni, "Clinically applicable machine learning approaches to identify attributes of chronic kidney disease (ckd) for use in low-cost diagnostic screening," IEEE Journal of Translational Engineering in Health and Medicine, vol. 9, pp. 1–11, 2021.

[10] M. A. R. Rahat, M. T. Islam, D. M. Cao, M. Tayaba, B. P. Ghosh, E. H. Ayon, N. Nobe, T. Akter, M. Rahman, and M. S. Bhuiyan, "Comparing machine learning techniques for detecting chronic kidney disease in early stage," Journal of Computer Science and Technology Studies, vol. 6, no. 1, pp. 20–32, 2024.

[11] R. W. Major, D. Shepherd, J. F. Medcalf, G. Xu, L. J. Gray, and N. J. Brunskill, "The kidney failure risk equation for prediction of end stage renal disease in uk primary care: an external validation and clinical impact projection cohort study," PLoS medicine, vol. 16, no. 11, p. e1002955, 2019.

[12] C. de Faria Rezende, S. V. B. Pinheiro, M. L. Cherchiglia, M. G. M. G. Penido, and H. A. da Rocha, “Epidemiology of pediatric chronic kidney disease and pediatric kidney failure: what we learn from the studies,” International Journal of Clinical Nephrology, vol. 5, no. 5, 2023.

[13] M. Wainstein, A. Rahimi, I. Katz, H. Healy, S. Pirabhahar, K. Turner, and S. Shrapnel, “A comparison between a random forest model and the kidney failure risk equation to predict progression to kidney failure,” medRxiv, 2023.

[14] P. V. Hartman, “Advances in ckd diagnosis and management,” Journal of Clinical Medicine, vol. 12, no. 5, pp. 478–490, 2021.

[15] J. Doe, “Artificial intelligence in medical diagnostics,” Health Informatics Journal, vol. 10, no. 4, pp. 254–265, 2020.

[16] R. Smith, “Ai in healthcare: Benefits and challenges,” Journal of Healthcare Informatics Research, vol. 14, no. 2, pp. 123–130, 2019.

[17] L. Zhang, “Machine learning for non-linear data in medicine,” Journal of Machine Learning Research, vol. 16, pp. 501–520, 2019.

[18] A. J. Aljaaf et al., “Machine learning for early detection of chronic kidney disease,” IEEE Access, vol. 6, pp. 7249–7257, 2018.

[19] H. Almasoud and T. Ward, “Feature selection for ckd detection using machine learning algorithms,” International Journal of Medical Informatics, vol. 124, pp. 55–65, 2019.

[20] A. Kumar et al., “Factors leading to ckd: A comprehensive study,” Nephrology Journal, vol. 11, no. 6, pp. 345–354, 2020.

[21] S. Lee et al., “Hypertension and diabetes as predictors of ckd,” Journal of Nephrology, vol. 22, no. 3, pp. 157–165, 2019.

[22] N. Tangri and T. Ferguson, “Practical utilization of prediction equations in chronic kidney disease,” Blood purification, pp. 1–7, June 2023.

[23] D. Dingwei and S. D. Woods, “A predictive model for progression of chronic kidney disease to kidney failure using a large administrative claims database,” ClinicoEconomics and Outcomes Research, vol. 13, pp. 475–486, 2021.

[24] N. Tangri and T. Ferguson, “Practical utilization of prediction equations in chronic kidney disease,” Blood purification, pp. 1–7, June 2023.

[25] V. S. R. Chennareddy, S. Tirunagari, S. Mohan, D. Windridge, and Y. Balla, “Interpretable chronic kidney disease risk prediction from clinical data using machine learning,” in International Conference on Multi-disciplinary Trends in Artificial Intelligence. Springer, 2023, pp. 683–691.

[26] V. Arumugham, B. Sankaralingam, U. Jayachandran, K. Krishna, S. Sundarraj, and M. Mohammed, "An explainable deep learning model for prediction of early-stage chronic kidney disease," Computational Intelligence, 2023.

[27] Y. Gu, Y. Gong, M. Wang, S. Jiang, C. Li, and Z. Yuan, "Enhancing kidney failure analysis: Web application development for longitudinal trajectory clustering," medRxiv, pp. 2023–05, 2023.

[28] S. Patel, R. Patel, N. Ganatra, S. Khant, and A. Patel, "An experimental study and performance analysis of supervised machine learning algorithms for prognosis of chronic kidney disease," in 2022 First International Conference on Electrical, Electronics, Information and Communication Technologies (ICEEICT). IEEE, 2022, pp. 1–6.

[29] T. Ferguson, P. Ravani, M. M. Sood, A. Clarke, P. Komenda, C. Rigatto, and N. Tangri, "Development and external validation of a machine learning model for progression of ckd," Kidney International Reports, vol. 7, no. 8, pp. 1772–1781, 2022.

[30] S. Wang, J. Han, S. Y. Jung, T. J. Oh, S. Yao, S. Lim, H. Hwang, H.-Y. Lee, and H. Lee, "Development and implementation of patient-level prediction models of end-stage renal disease for type 2 diabetes patients using fast healthcare interoperability resources," Scientific Reports, vol. 12, no. 1, p. 11232, 2022.

[31] V. Arumugham, B. P. Sankaralingam, U. M. Jayachandran, K. V. S. S. R. Krishna, S. Sundarraj, and M. Mohammed, "An explainable deep learning model for prediction of early-stage chronic kidney disease," Computational Intelligence, 2023.

[32] P. A. Moreno-Sanchez, "An explainable classification model for chronic kidney disease patients." CoRR, 2021.

[33] S. Patel, R. Patel, N. Ganatra, S. Khant, and A. Patel, "An experimental study and performance analysis of supervised machine learning algorithms for prognosis of chronic kidney disease," 2022.

[34] G. Nandhini and J. Aravinth, "Chronic kidney disease prediction using machine learning techniques," 2021.

[35] R. H. Khan, J. Miah, M. A. R. Rahat, A. H. Ahmed, M. A. Shahriyar, and E. R. Lipu, "A comparative analysis of machine learning approaches for chronic kidney disease detection," 2023.

[36] A. Farjana, F. T. Liza, P. P. Pandit, M. C. Das, M. Hasan, F. Tabassum, and M. H. Hossen, "Predicting chronic kidney disease using machine learning algorithms," 2023.

[37] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining, 2016, pp. 785–794.

[38] C. Wohlin, M. Höst, and K. Henningsson, Empirical Research Methods in Software Engineering. Declarative Agent Languages and Technologies X, 2003, p. 7–23.

[39] M. S. Arif, A. Mukheimer, and D. Asif, "Enhancing the early detection of chronic kidney disease: A robust machine learning model," Big Data and Cognitive Computing, vol. 7, no. 3, p. 144, 2023.

[40] S. J. Maisha, E. Biswangri, M. S. Hossain, and K. Andersson, "An approach to detect chronic kidney disease (ckd) by removing noisy and inconsistent values of uci dataset," in Proceedings of the Third International Conference on Trends in Computational and Cognitive Engineering: TCCE 2021. Springer, 2022, pp. 457–472.

[41] S. Iimori, S. Naito, Y. Noda, H. Sato, N. Nomura, E. Sohara, T. Okado, S. Sasaki, S. Uchida, and T. Rai, "Prognosis of chronic kidney disease with normal-range proteinuria: The ckd-route study," PLOS ONE, vol. 13, no. 1, p. e0190493, 2018. [Online]. Available: https://dx.doi.org/10.1371/journal.pone.0190493

[42] A. S. Levey, L. A. Stevens, C. H. Schmid, Y. L. Zhang, A. F. Castro, H. I. Feldman, J. W. Kusek, P. Eggers, F. Van Lente, T. Greene, and et al., "A new equation to estimate glomerular filtration rate," Annals of Internal Medicine, vol. 150, no. 9, p. 604, 2009. [Online]. Available: https://dx.doi.org/10.7326/0003-4819-150-9-200905050-00006

[43] S. A. Hicks, I. Strümke, V. Thambawita, M. Hammou, M. A. Riegler, P. Halvorsen, and S. Parasa, "On evaluation metrics for medical applications of artificial intelligence," Scientific Reports, vol. 12, no. 1, 2022.

[44] M. Wainstein, A. K. Rahimi, I. Katz, H. Healy, S. Pirabhahar, K. Turner, and S. Shrapnel, "A comparison between a random forest model and the kidney failure risk equation to predict progression to kidney failure," medRxiv, 2023. [Online]. Available: https://www.medrxiv.org/content/early/2023/05/17/2023.05.16.23290068