KIDNEY DISEASES PREDICTION USING MACHINE LEARNING

Minor project-II report submitted
in partial fulfillment of the requirement for award of the degree of

Bachelor of Technology
in
Computer Science & Engineering

By

K. MOHITH SAI	(21UEDS0038)	**(VTU 19278)**
THANGA VISHNU VARDHAN REDDY	(21UEDS0062)	**(VTU 19696)**
KALAVAGUNTA SAKETH	(21UEDS0029)	**(VTU 19277)**

Under the guidance of
Dr. C. KOTTESWARAN, M.E,PhD
ASSOCIATE PROFESSOR

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
SCHOOL OF COMPUTING

VEL TECH RANGARAJAN DR. SAGUNTHALA R&D INSTITUTE OF SCIENCE & TECHNOLOGY

(Deemed to be University Estd u/s 3 of UGC Act, 1956)
Accredited by NAAC with A++ Grade
CHENNAI 600 062, TAMILNADU, INDIA

May, 2024

KIDNEY DISEASES PREDICTION USING MACHINE LEARNING

Minor project-II report submitted
in partial fulfillment of the requirement for award of the degree of

Bachelor of Technology
in
Computer Science & Engineering

By

K. MOHITH SAI	(21UEDS0038)	**(VTU 19278)**
THANGA VISHNU VARDHAN REDDY	(21UEDS0062)	**(VTU 19696)**
KALAVAGUNTA SAKETH	(21UEDS0029)	**(VTU 19277)**

Under the guidance of
Dr. C. KOTTESWARAN, M.E,PhD
ASSOCIATE PROFESSOR

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
SCHOOL OF COMPUTING

VEL TECH RANGARAJAN DR. SAGUNTHALA R&D INSTITUTE OF SCIENCE & TECHNOLOGY

(Deemed to be University Estd u/s 3 of UGC Act, 1956)
Accredited by NAAC with A++ Grade
CHENNAI 600 062, TAMILNADU, INDIA

May, 2024

CERTIFICATE

It is certified that the work contained in the project report titled ”KIDNEY DISEASES PREDICTION USING MACHINE LEARNING ” by ”K. MOHITH SAI (21UEDS0038) ,THANGA VISHNU VARDHAN REDDY (21UEDS0062), KALAVAGUNTA SAKETH (21UEDS0029),” has been carried out under my supervision and that this work has not been submitted elsewhere for a degree.

Signature of Supervisor
Computer Science & Engineering
School of Computing
Vel Tech Rangarajan Dr. Sagunthala R&D
Institute of Science & Technology
May, 2024

Signature of Professor In-charge
Computer Science & Engineering
School of Computing
Vel Tech Rangarajan Dr. Sagunthala R&D
Institute of Science & Technology
May, 2024

DECLARATION

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

K. MOHITH SAI

Date: / /

THANGA VISHNU VARDHAN REDDY

Date: / /

KALAVAGUNTA SAKETH

Date: / /

APPROVAL SHEET

This project report entitled KIDNEY DISEASES PREDECTION USING MACHINE LEARNING by K. MOHITH SAI (21UEDS0038), THANGA VISHNU VAEDHAN REDDY (21UEDS0062), KALAVGUNTA SAKETH (21UEDS0029) is approved for the degree of B.Tech in Computer Science & Engineering.

Examiners **Supervisor**

Dr. C. KOTTESWARAN, M.E,PhD,

Date: / /

Place:

ACKNOWLEDGEMENT

We express our deepest gratitude to our respected **Founder Chancellor and President Col. Prof. Dr. R. RANGARAJAN B.E. (EEE), B.E. (MECH), M.S (AUTO),D.Sc., Foundress President Dr. R. SAGUNTHALA RANGARAJAN M.B.B.S.** Chairperson Managing Trustee and Vice President.

We are very much grateful to our beloved **Vice Chancellor Prof. S. SALIVAHANAN,** for providing us with an environment to complete our project successfully.

We record indebtedness to our **Professor & Dean, Department of Computer Science & Engineering, School of Computing, Dr. V. SRINIVASA RAO, M.Tech., Ph.D.,** for immense care and encouragement towards us throughout the course of this project.

We are thankful to our **Head, Department of Computer Science & Engineering, Dr.M.S. MURALI DHAR, M.E., Ph.D.,** for providing immense support in all our endeavors.

We also take this opportunity to express a deep sense of gratitude to our **Internal Supervisor Dr.C.KOTTESWARAN, M.E.,** for his cordial support, valuable information and guidance, he helped us in completing this project through various stages.

A special thanks to our **Project Coordinators Mr. V. ASHOK KUMAR, M.Tech., Ms. C. SHYAMALA KUMARI, M.E., Mr. SHARAD SHANDHI RAVI, M.Tech.,** for their valuable guidance and support throughout the course of the project.

We thank our department faculty, supporting staff and friends for their help and guidance to complete this project.

K. MOHITH SAI	**(21UEDS0038)**
THANGA VISHNU VAEDHAN REDDY	**(21UEDS0062)**
KALAVGUNTA SAKETH	**(21UEDS0029)**

ABSTRACT

Chronic Kidney Disease is a serious lifelong condition that induced by either kidney pathology or reduced kidney functions. According to the survey in 2021-2022 for every 10 in 3, people are suffering from this chronic kidney disease. It enables us to introduce the optimal subset of parameters to feed machine learning to build a set of predictive models. K-Nearest Neighbor, Random Forest, LBGM and Decision Tree methods are applied for prediction. Among these four methodologies, the proposed model suggests an LBGM is suitable for the early prediction of this kind of disease. Performance measures show that Light Gradient Boosted Machine (LGBM) gives 99 98 procedure concludes that advances in machine learning and predictive analytics, represent a promising model to recognize intelligent solutions, which in turn prove the ability of predication in kidney disease.

Keywords: Chronic Kidney Disease, Decision Tree, K-Nearest Neighbor, Light Gradient Boosted Machine, Machine Learning, Prediction, Random Forest.

LIST OF FIGURES

LIST OF ACRONYMS AND ABBREVIATIONS

SI.NO	ABBREVIATION	DEFINITION
1	AI	Artificial Intelligence
2	CKD	Chronic Kidney Disease
3	CNN	Convolutional Neural Network
4	DT	Decision Tree
5	ESRD	End Stage Renal Disease
6	FP	False Positive
7	FN	False Negative
8	GBF	Gradient Boosting Framework
9	GFR	Glamorise Filtration Rate
10	KNN	K-Nearest Neighbor
11	LGBM	Light Gradient Boosted Machine
12	RF	Random Forest
13	SVM	Support Vector Machine
14	TP	True Positive
15	TN	True Negative
16	ML	Machine Learning
17	UCI	Unique Client Identifier

TABLE OF CONTENTS

Chapter 1

INTRODUCTION

1.1 Introduction

Chronic kidney disease (CKD) is a significant public health problem for worldwide, especially for a low and medium-income countries. CKD means that the kidney does not work and cannot correctly filter the blood. About 10 percent of the population for a worldwide suffering from (CKD), and millions of die each year because of they couldn't get the affordable treatment, with the number increasing in the elderly.

In 2021, a study was conducted by International Society of Numerology(ISN) on global burden disease, they reported that CKD has been raised an important cause of the mortality worldwide with the number of deaths increasing by 82.3 percent in the last two decades. Also, the number of patients reaching End-Stage Renal Disease (ESRD) is increasing, which requires kidney transplantation or dialysis to save the patient‘s lives.

The worst possible outcome of the chronic kidney disease and the symptoms causing any reduced kidney functioning would lead to kidney failure. When the symptoms are become severe and uncontrollable they can be treated through the dialysis and transplantation. Early detection and treatment of CKD can slow or stop the progression of the kidney disease.But the CKD, in it's early stages, has no symptoms.

The Proper diagnosis and the testing may be the only way to find out whether the patient has been affected by kidney disease. Early detection of CKD in its initial stages can help the patient get the effective treatment and then prohibit the progression to ESRD. CKD is a Long term condition induced the damage done to both kidneys. Kidney damage refers to any kind of kidney pathology that gives the possibility to reduce the capacity of kidney functions, particularly the reduction in Glamorise Filtration Rate (GFR).

Chronic kidney disease (CKD) is a progressive condition that affects the kidneys, which are responsible for filtering waste products and excess fluids from the blood. CKD can be caused by a range of factors including high blood pressure, diabetes, autoimmune diseases, and inherited genetic conditions. As CKD progresses, the kidneys become less able to perform their filtering function, leading to a buildup of waste products and fluids in the body. This can cause a range of symptoms including fatigue, weakness, nausea, and swelling in the legs and feet.

1.2 Aim of the project

The main aim is to identify whether a particular patient is affected by CKD (or) not and it has to be accurate and precise. So, for that we are going to purpose a correlate four pre-existing Machine Learning Algorithms to find the best among all. For this purpose, we gathered a CKD data set from UCI machine learning repository and we examining the correlation between the development of the CKD and predictors using a predictive approach of the analysis. This will help us to reduce the number of required parameters to predict the CKD disease occurrence as well as eliminating the missing, redundant and noisy data. To use certain features to measure its accuracy and predictions.

1.3 Project Domain

Machine learning (ML) is a subfield of artificial intelligence (AI) that focuses on the development of algorithms and models that can learn and make predictions or decisions based on patterns found in data. In the context of the project Kideny Disease Prediction, machine learning techniques is used to analyze large datasets of patient records, and other relevant information to identify patterns that may indicate the presence of Kideny Disease or the likelihood of developing the disease. The algorithms such as K-Nearest Neighbor,Random Forest,Decision Tree,Light Gradient Boosted Machine are used to train models on labeled data to classify patients as either having kideny disease or not. The framework utilizes a machine learning algorithm called Light Gradient Boosted Machine to analyze patient data and make predictions about their risk of developing kideny disease.

Light Gradient Boosted Machine is a powerful algorithm that has been used successfully in several medical domains. It works by building multiple decision trees and combining their predictions to obtain a more accurate result. The project domain includes data collection, processing, and analysis. The framework will be trained on a large dataset of medical images, patient history, and demographic information to learn the patterns and characteristics of kideny disease.

1.4 Scope of the Project

The scope of the project is to Predict whether the patient is suffering from kidney diseases or not by using machine learning algorithms. They can minimize the risk caused by the diseases because “prevention is better than cure”.

Chapter 2

LITERATURE REVIEW

[1] Imesh Udara Ekanayake,et.al.,(2020) , have proposed that the workflow to predict CKD status based on clinical data, incorporating data prepossessing, a missing value handling method with collaborative filtering and attributes selection. The data distribution has properly covered the whole domain in CKD, but the general attributes like appetite, anaemia and pedal oedema are biased towards CKD. It easy to achieve an accurate prediction using the KNN algorithm in machine learning with the data set. It highlights the importance of incorporating the domain knowledge into feature selection when analysing clinical data related to CKD. In the general context, to achieve the accuracy may lead to false positives as observed in Recall.

[2] Yedilkhan Amirgaliyev,et.al.,(2019) , have identified that there are less studies performed in CKD by using SVMs. But there are several studies where SVM has been used abundantly. They used SVMs as a classification model for detecting and diagnosing malignant and benign tumors based on MRI features, ultrasound feature and mammographic features. This result tries to create and test SVM classifier over CKD dataset. To improve the accuracy of CKD classification as positive and negative, the performance of Support Vector Machine (SVM) was evaluated. The SVM is a flexible classifier algorithm that has been proposed as an effective statistical learning method for pattern recognition, which is based on finding optimal hyperplane to separate different classes mapping input data into higher-dimensional feature space. It is automatic classification algorithm for kidney disease diagnosis based on clinical history, physical examinations, and laboratory tests, which are noninvasive, cheap and safe.

[3] Khalid Twarish Alhamazani,et.al.,(2021) , represented the CRISP-DM methodology to the context of the problem so that the different logically organized stages were taken, data collection, pre-processing, learning, evaluation, and selection, which allowed the construction of a model capable of classifying the possibility of a diagnosis of CKD with an accuracy of 93 forests algorithm has obtained quite

optimal results, where predictions of 93been obtained. The innovation of this work results from the design adjusted to the environment of the health system in Iraq and the pathology of the CKD in our country, with a methodology adapted to the case study and a production architecture proposal for the model with Microsoft Azure tools of form that allows satisfying in the future the scalability of the solution.

[4] Pankaj Chittora,et.al., (2021) , have developed a model to predict CKD disease in patients. The performance of the model was tested on both all attributes and selected features. Among feature selection methods there were Wrapper, Filter and Embedded allowing to select vital features. Classifier algorithms performance was tested on the selected features. The machine learning classifiers such as logistic regression and K- nearest neighbors (KNN) were used for training the model. Logistic and KNN classifiers give satisfactory result and have the negligible difference between precision and recall values. Logistic and KNN algorithms have not enough capacity to distinguish between positive class and negative class as the related score is very low.

[5] Jing Xiao,et.al., (2019) , have proposed the work to predict the severity of CKD using more easily available demographic and blood biochemical features during follow-up. Developed and compared several predictive models using statistical, machine learning and neural network approaches. A Web tool (CKD Prediction System) for clinical practice that can be widely used in the evaluation of proteinuria progress in nephrology has developed. The comparison in this study shows that the linear models performed better. Journal of Translational Medicine Predicting the severity of CKD using machine learning tools. The dialysis machine mixes and monitors the dialysate. Dialysate is the fluid that helps remove the unwanted waste. The incremental feature selection approach with Convolutional Neural Network (CNN) is applied to anticipate disease.

[6] Janakiram Ampolu,et.al., (2019) , represented the causes , symptoms, and complications of CKD in a clear manner such that even common people can easily understand. Once a patient is detected and proved to be affected by CKD then the patient as well as the care takers, including doctors, must follow some constraints. There by it is possible to prevent CKD progression in the patient. Modern methods are needed to prevent the pathogens which are responsible for CKD. With the help

of various engineering techniques one can easily design controllers to assess as well as to prevent CKD permanently. The easiest procedure for identifying CKD is to screen people. Current recommendations suggest screening of individuals with diabetes, hypertension, cardiovascular disease, and family history of kidney diseases in the course of routine health checkups. Much work has been done in medical sciences in the area of CKD, but there is still scope for further research. From the recent studies, advanced tools such as data mining, etc., are considered to be the current trend in the area of CKD.

[7] Salma Shaji,et.al., (2020) , have proposed Chronic Kidney Disease also recognized as Chronic Renal Disease, is an uncharacteristic functioning of kidney or a failure of renal function expanding over a period of months or years. Habitually, chronic kidney disease is detected during the screening of people who are known to be in threat by kidney problems, such as those with high blood pressure or diabetes and those with a blood relative Chronic Kidney Disease(CKD) patients. So the early prediction is necessary in combating the disease and to provide good treatment. This study proposes the use of machine learning techniques for CKD such as Ant Colony Optimization(ACO) technique and Support Vector Machine(SVM) classifier. Final output predicts whether the person is having CKD or not by using minimum number of features. CKD is caused due to diabetes and high blood pressure. Due to Diabetes our many organs get affected and it will be followed by high blood sugar. So it is important to predict the disease as early as possible. This study improvises some of the machine learning techniques to predict the disease.

[8] Revathy Ramesh,et.al., (2020) , proposes a datamining framework for knowledge discovery on the CKD datasets. Large amounts of CKD datasets are collected. Data preperation and preprocessing is done using the traditional methods of data mining process. Three machine learning algorithms namely Decision tree, Random Forest and Support Vector machines are used to predict the early occurence of CKD. The goodness of each algorithm is analysed.The biosciences have advanced to a larger extent and have generated large amounts of information from Electronic Health Records. This have given rise to the acute need of knowledge generation from this enormous amount of data. Data mining methods and machine learning play a major role in this aspect of biosciences. Chronic Kidney Disease(CKD) is a condition in which the kidneys are damaged and cannot filter blood as they always

do. A family history of kidney diseases or failure, high blood pressure, type 2 diabetes may lead to CKD. This is a lasting damage to the kidney and chances of getting worser by time is high. The very common complications that results due to a kidney failure are heart diseases, anemia, bone diseases, high potasium and calcium. The worst case situation leads to complete kidney failure and necessitates kidney transplant to live. An early detection of CKD can improve the quality of life to a greater extent. This calls for good prediction algorithm to predict CKD at an earlier stage . Literature shows a wide range of machine learning algorithms employed for the prediction of CKD. It uses data preprocessing,data transformation and various classifiers to predict CKD and also proposes best Prediction framework for CKD. The results of the framework show promising results of better prediction at an early stage of CKD.

[9] B.Sasi Varna,et.al., (2021) , have proposed several machine learning models in this section, the component models that performed better when diagnosing the data samples were chosen for inclusion. Examination of the component models' mistakes determined the component models' roles as potential biomarkers. Chronic kidney disease (CKD) is a global health issue that causes a high incidence of morbidity and death, as well as the onset of additional illnesses. Because there are no clear symptoms in the early stages of CKD, people frequently miss it. Early identification of CKD allows patients to obtain prompt therapy to slow the disease's development. Due of their rapid and precise identification capabilities, machine learning models can successfully assist doctors in achieving this aim.The CKD data set was collected from the machine learning repository at the University of California, Irvine (UCI). As a result, it will determine whether or not a patient has CKD and, if so, whether or not further drugs should be taken. Six machine learning algorithms (Logistic Regression, AdaBoost, Random Forest, Decision Tree, and Gradient Boosting) were used to establish models.

[10] Dibaba Adeba Debal,et.al., (2022) , represented with both binary and multi classifcation for stage prediction have been carried out. The prediction models used include Random Forest (RF), Support Vector Machine (SVM) and Decision Tree (DT). Analysis of variance and recursive feature elimination using cross validation have been applied for feature selection. Evaluation of the models was done using tenfold cross-validation. The results from the experiments indicated that RF based

on recursive feature elimination with cross validation has better performance than SVM and DT. Predictive analysis using machine learning techniques can be helpful through an early detection of CKD for efficient and timely interventions. In this study, Random Forest (RF), Support Vector Machine (SVM) and Decision Tree (DT) have been used to detect CKD. Most of previous researches focused on two classes, which make treatment recommendations difficult because the type of treatment to be given is based on the severity of CKD.

Chapter 3

PROJECT DESCRIPTION

3.1 Existing System

In machine learning, everyone tried to go with the best algorithm which will try to give a more accuracy and play a prominent role in prediction. It uses machine learning algorithms, such as random forests and gradient boosting. It discusses various machine learning approaches used for predicting chronic kidney disease, including decision trees, logistic regression, artificial neural networks, and support vector machines. It provides a comprehensive overview of the existing systems. To predict the outcomes of kidney transplants based on donor and recipient characteristics, Which are available to us, and by comparison, we might be able to notice it's compatibility and it's versatility in the model as well as the system. So, it is always better to look for a alternative approach. The advantages like Early detection and Prevention,Better treatment. Predicting kidney disease is a cost-effective way of identifying individuals who are at risk of developing the disease. Early detection and treatment can help prevent costly and invasive treatments in the future. The prediction of kidney disease can help to raise awareness of the disease and promote healthy lifestyles that can help reduce the risk of developing the disease. This can ultimately lead to improved public health outcomes

3.2 Proposed System

It is better to search for an alternative approach with promising results and we are going to do that. The proposed model compares four very commonly known machine learning algorithms and are going to state the best among them using a case study on Chronic Kidney disease Prediction. As stated, the four machine learning algorithms are: Random Forest, Decision Tree, LGBM, and K- Near Neighbour(KNN). The disadvantages of Chronic kidney disease often has no symptoms in the early stages,

and symptoms may not appear until the disease is advanced. This makes it difficult to diagnose and treat early. CKD can lead to other serious health problems, such as heart disease, high blood pressure, anemia, nerve damage, and bone disease. The advantage is early detection of kidney disease can help healthcare professionals to develop a personalized treatment plan that is tailored to the individual patient's needs. This can help improve treatment outcomes and reduce the risk of complications and by identifying the risk factors associated with kidney disease, individuals can take preventative measures to reduce their risk of developing the disease. These measures may include lifestyle changes such as adopting a healthy diet and regular exercise.

3.3 Feasibility Study

3.3.1 Economic Feasibility

The kidney disease prediction using machine learning project can be economically feasible if the potential benefits out weigh the costs. Proper planning and assessment of the project costs and benefits can help ensure its economic viability.

3.3.2 Technical Feasibility

The kidney disease prediction using machine learning project can be technically feasible if there is access to high-quality data, appropriate machine learning algorithms, powerful computing resources, reliable deployment infrastructure, and ongoing monitoring and maintenance.

3.3.3 Social Feasibility

The social feasibility of a kidney disease prediction using machine learning project depends on the acceptance and adoption of the technology by patients, healthcare providers, and other relevant parties. The project must comply with ethical standards and regulations and be designed to be accessible, equitable, and aligned with the goals of the healthcare system.

3.4 System Specification

3.4.1 Hardware Specification

- Processor : Intel(R) Core(TM) i3-10110U CPU @ 2.10GHz 2.60 GHz
- RAM : 8.00 GB (7.78 GB usable)
- Hard Disc Space : 8GB

3.4.2 Software Specification

- Operating System : Windows 7,8,10 (or) Mac (or) Linux
- Coding Language : Python - version(3.5)
- Platform Requirement: Google Colab
- Framework: Kara's (or) Tensor flow (or) Kaggle

3.4.3 Standards and Policies

General Data Protection Regulation (GDPR):
GDPR is a regulation of the European Union that came into effect in May 2018. It sets out strict guidelines for the processing of personal data, including health data. If you are collecting data from patients in the EU, you must comply with GDPR.
Intellectual Property Rights (IPR):
An IRB is an independent committee that reviews and approves research studies involving human subjects. IRBs ensure that the study is conducted ethically and that the privacy and rights of the patients are protected.

Chapter 4

METHODOLOGY

4.1 General Architecture

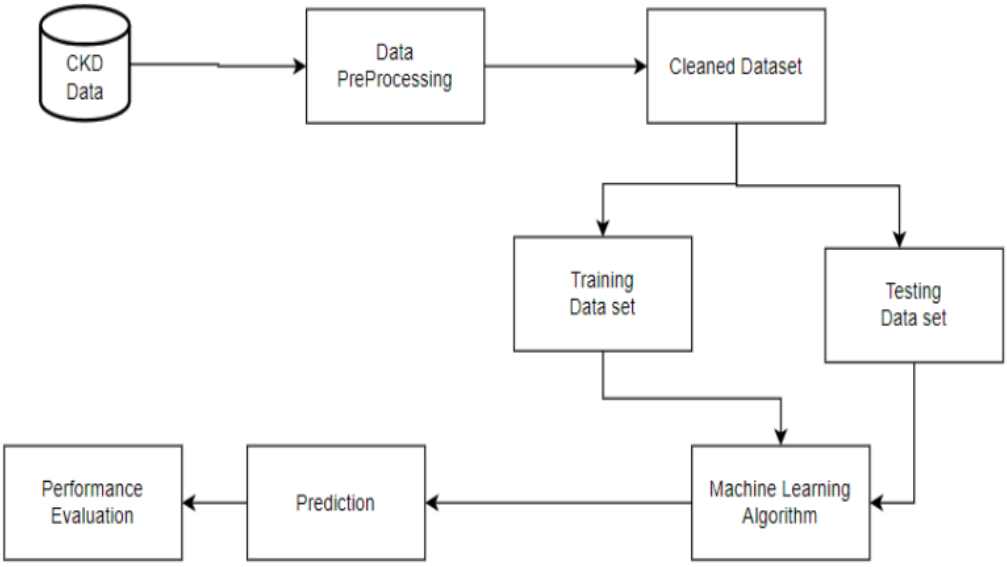


Figure 4.1: **CKD-Architecture Diagram**

In figure 4.1 show the architecture of the kideny disease prediction. Intially the data set is collected from the University of California, Irvine (UCI). Then that data set is preprocessed to remove the null values and repeated values. Then the output is cleaned dataset then that particular dataset is divided into two different types like training dataset and testing dataset it under goes to the machine learning algorithms like K-Nearest Neighbor,Random Forest,Decision Tree and Light Gradient Boosted Machine. By using these four algorithms the disesase is predicted and performance evaluation is done.

4.2 Design Phase

4.2.1 Data Flow Diagram

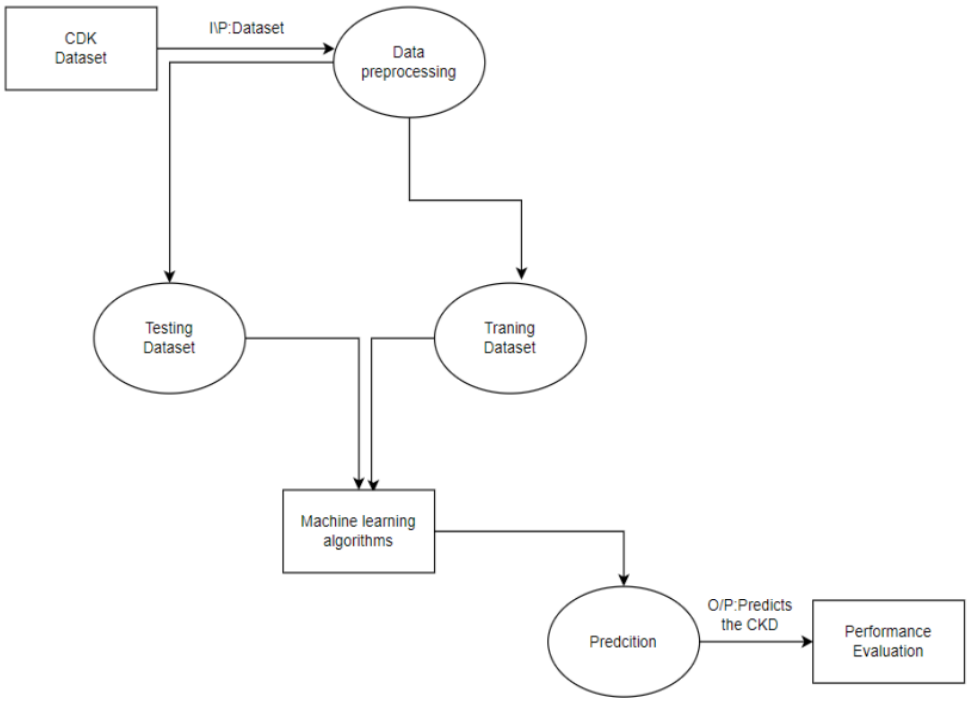


Figure 4.2: **CKD-Data flow Diagram**

In Figure4.2 Data flow diagram for chronic kidney disease classification using machine learning algorithms. The training dataset and testing dataset are used in the machine learning algorithms for predicting the CKD.Based on the performance evaluation, the kidney disease can be predicted. It has experimented on UCI dataset which contains early stages of 400 CKD patients with 25 attributes.

4.2.2 Use Case Diagram

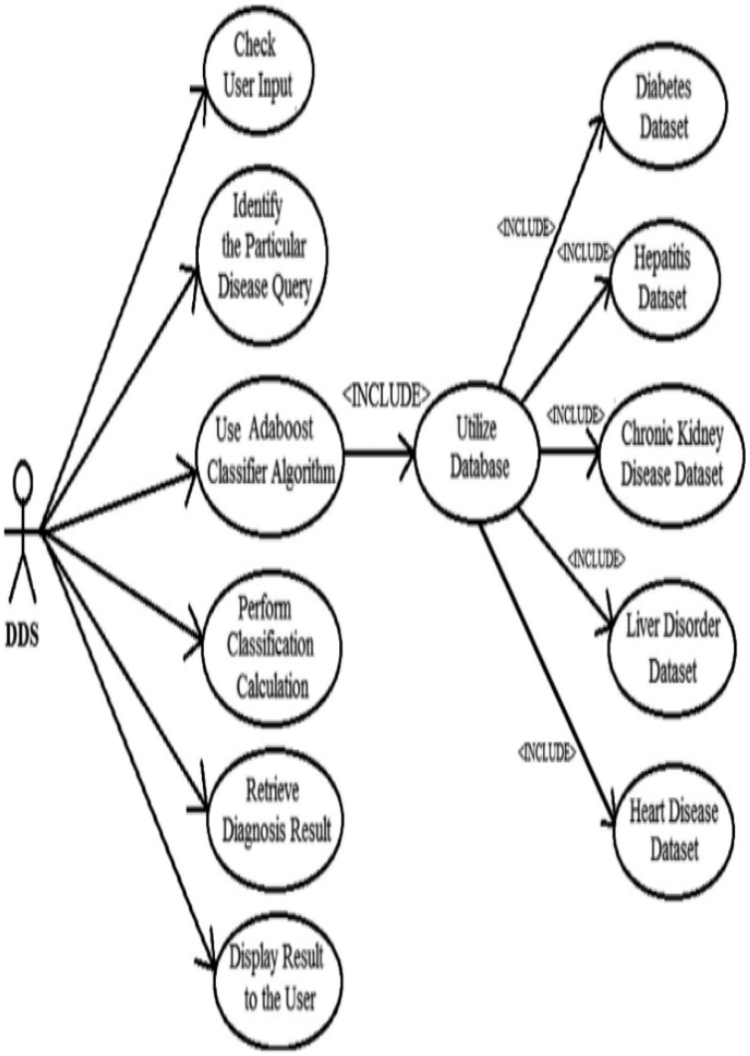


Figure 4.3: **CKD-Usecase Diagram**

In figure 4.3 Use case diagram is a type of behavioral diagram defined by and created from a Usecase analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what systems functions are performed for which actor. The user interacts with the system by providing patient data and viewing the prediction results. The system performs the data preprocessing, feature selection, machine learning model training, and prediction. This use case diagram provides an overview of the functionalities of a kidney disease prediction system using machine learning and how users can interact with it.

4.2.3 Class Diagram

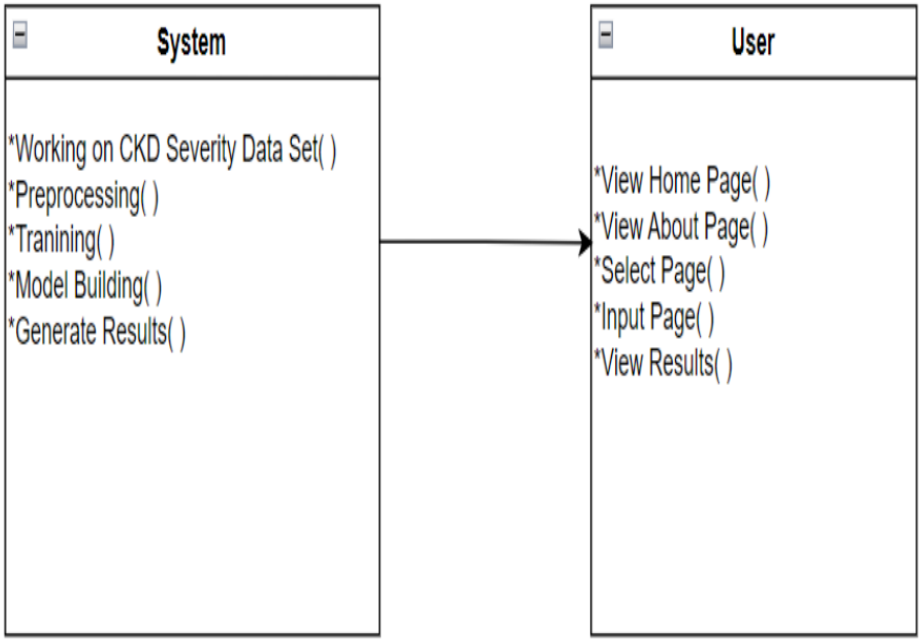


Figure 4.4: **CKD-Class Diagram**

In figure 4.4 Class Diagram is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and The relationships between these classes can be shown using associations, where one class is connected to another class. For example, the User class can have an association with the Prediction class to represent that the user can view the prediction results. Similarly, the Patient class can have an association with the Data Preprocessing class, Feature Selection class, Machine Learning Model class, and Prediction class to represent the flow of data and functionalities.This class diagram provides a static structure of a kidney disease prediction system using machine learning and shows how the different classes are related to each other.

4.2.4 Sequence Diagram

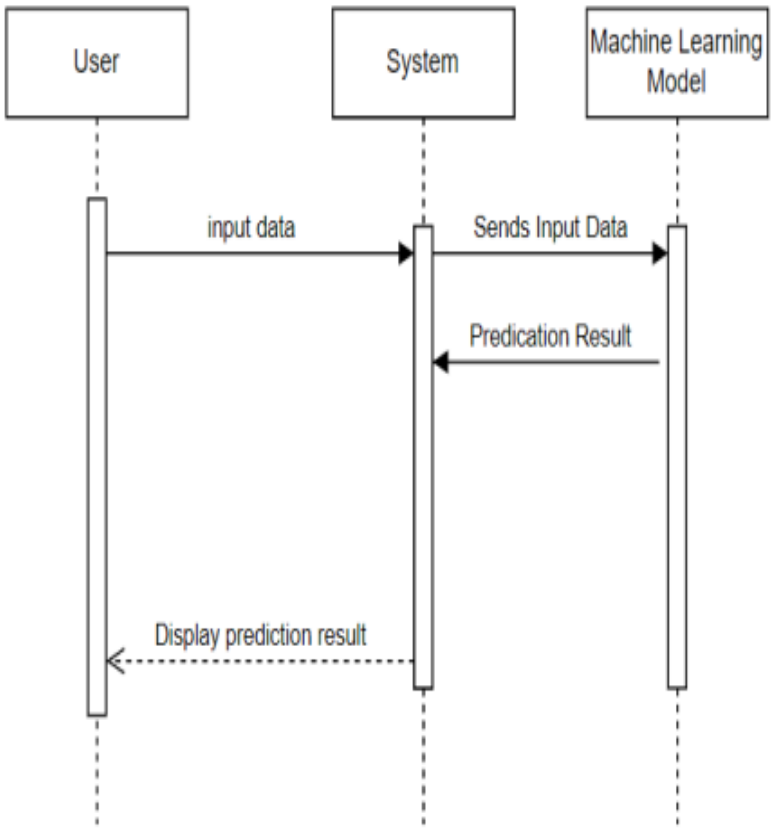


Figure 4.5: **CKD-Sequence Diagram**

In figure 4.5 Sequence Diagram is a kind of interaction diagram that shows how processes operate with one another and in what order. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.The sequence diagram shows the flow of messages between the objects or components involved in the system. The user interacts with the system by providing patient data and viewing the prediction results. The system performs the data preprocessing, feature selection, machine learning model training, and prediction. The sequence diagram shows the order in which these interactions occur.This sequence diagram provides a dynamic view of a kidney disease prediction system using machine learning and shows how the different components interact with each other during the prediction process.

4.2.5 Activity Diagram

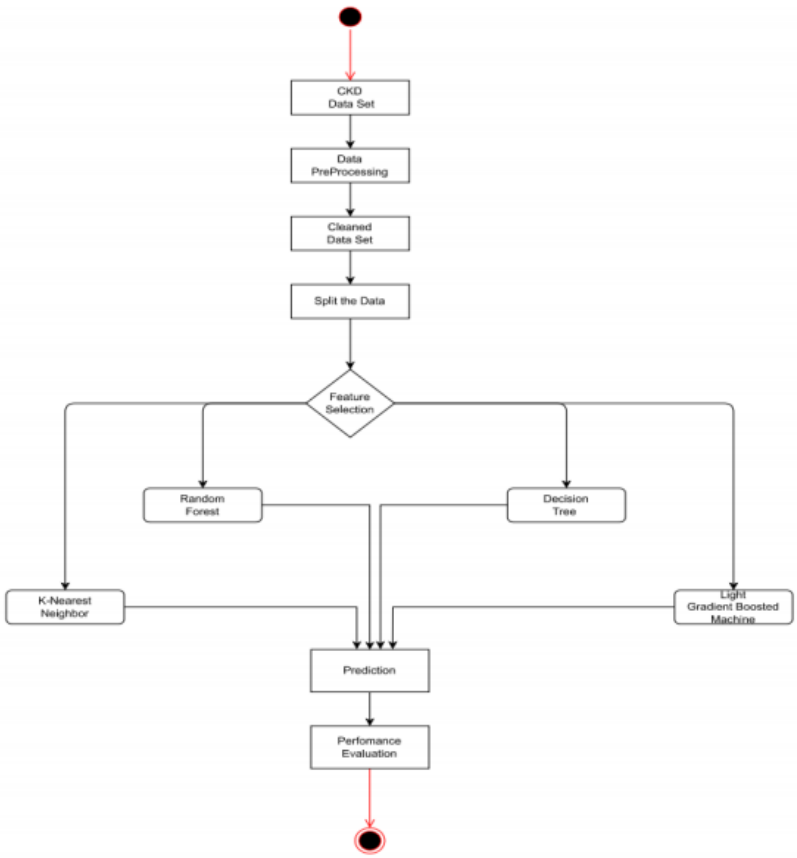


Figure 4.6: **CKD-Activity Diagram**

In figure 4.6 graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modelling Language, activity diagrams can be used to describe the business and operational stepby-step workflows of components in a system. An activity diagram shows the overall flow of control.

4.3 Algorithm & Pseudo Code

4.3.1 Algorithm

1. Import the LGBMClassifier from the lightgbm library
• from lightgbm import LGBMClassifier
2. Initialize the LGBMClassifier model with a learning rate of 1
• lgbm = LGBMClassifier(learning rate = 1)

3. Train the model on the training data (x train and y train)
• lgbm.fit(x train, y train)
4. Evaluate the performance of the model on the test data (x test and y test)
• lgbm acc = accuracy score(y test,lgbm.predict(x test))
5. Print the training accuracy, test accuracy, confusion matrix, and classification report of the LGBMClassifier model
• print the Training Accuracy of LGBM Classifier is accuracy score(y train, lgbm.predict(x train))
• print the Test Accuracy of LGBM Classifier is lgbm acc
• print the confusion matrix(y test, lgbm.predict(x test))
• print the classification report(y test, lgbm.predict(x test))
6. Create a pandas DataFrame to compare the performance of different models
• models = pd.DataFrame('Model' : ['KNN', 'Decision Tree Classifier', 'LGBM Classifier'], 'Score' : [knn acc, dtc acc, lgbm acc])
7. Sort the models DataFrame by the 'Score' column in descending order
• models.sort values(by = 'Score', ascending = False)

4.3.2 Pseudo Code

```

# Import necessary libraries
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score

# Load the dataset
data = pd.read_csv('kidneydisease.csv')

# Data cleaning and preprocessing
# ...

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Feature scaling
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)

```

```
# Train the logistic regression model
model = LogisticRegression(random_state=42)
model.fit(X_train, y_train)

# Make predictions on the testing set
y_pred = model.predict(X_test)

# Evaluate the model's accuracy
accuracy = accuracy_score(y_test, y_pred)
print('Accuracy:', accuracy)
```

4.4 Module Description

4.4.1 Data Collection and Training

1. **Data Collection**: The Dataset of prediction of chronic kidney disease using machine learning algorithm is downloaded from UCI repository. In that dataset there are 400 patient records are included. Also they include 25 attributes but we take only 14 attributes for building model. Age, Blood pressure, Albumin, Red blood cells, Pus cell, Pus cells clumps, Serum creatinine, Haemoglobin, White blood cell count, Red blood cell count, Anaemia, Classification, Appetite, Packed cell volume all this 14 attributes are used to build model.

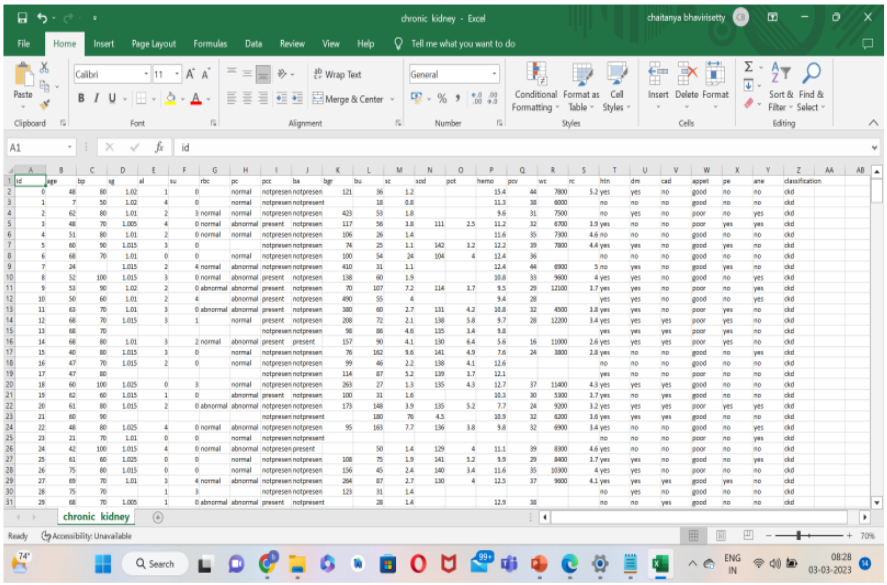


Figure 4.7: **CKD Data Set**

2. **Preprocessing**: Data Cleaning: Gather open source raw data of CKD patients available on internet. Data obtained from internet does not contains the name of the attribute so first we assigned the names to the attribute. Missing values in the dataset like NA's or blank values are removed by using WEKA function "ReplaceMissingValues" used, which replaces NA's with the mean values of that attribute. Data Reduction: Out of 25 attributes present in the dataset, we have selected 14 important attributes required to build predictive model.

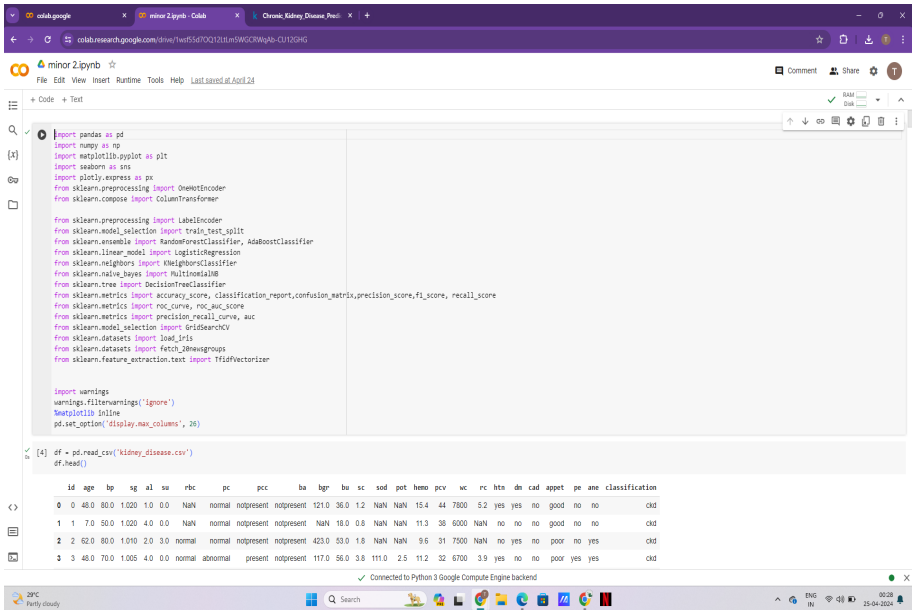


Figure 4.8: **CKD Data Preprocessing**

3. **Machine Learning Model**: The Model Development is a developing the machine learning model for kidney disease prediction requires careful consideration of the data, features, and model selection. It's important to ensure the model is accurate, reliable, and scalable for real-world applications and we apply the four types of machine learning algorithms like K-NN, RF, LGBM, Decision Tree.Finding the more accuracys out of those four algorithms.

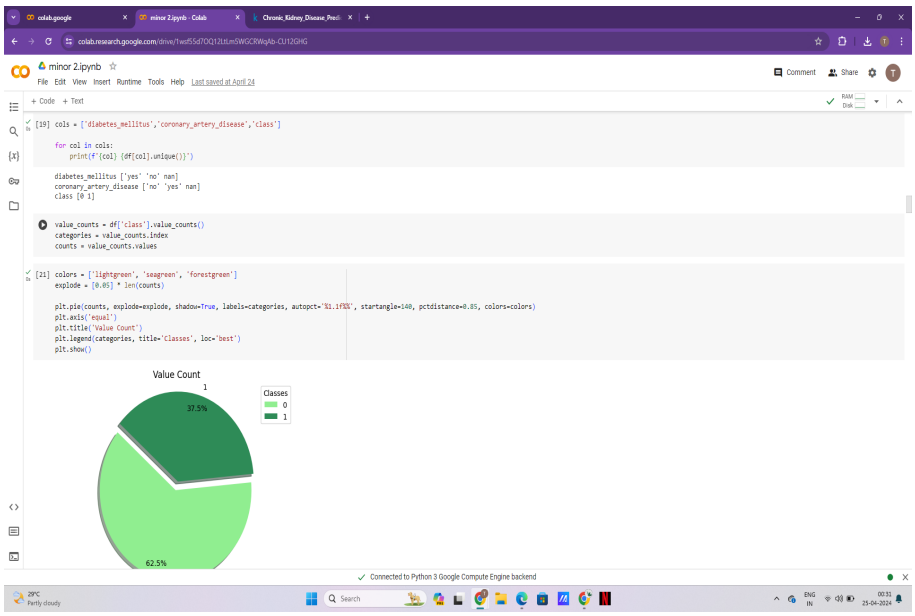


Figure 4.9: **Code development**

4.4.2 Graphical User Interface for the Application

1. **Input Fields**:
Age: Textbox or dropdown menu for entering the patient's age.
Gender: Radio buttons or dropdown menu for selecting male/female.
Blood Pressure: Textbox for systolic and diastolic blood pressure readings.
Blood Glucose: Textbox for entering blood glucose levels.
Serum Creatinine: Textbox for entering serum creatinine levels.
Albuminuria: Textbox for entering albuminuria levels.
Family History: Checkbox for indicating family history of CKD.
2. **Predict Button**:Button to trigger the prediction process based on the input data.
3. **Output Display**: Text area or label to display the predicted CKD risk or classification (e.g., low risk, moderate risk, high risk).
4. **Reset Button**: Button to clear all input fields and reset the interface.
5. **Feedback Mechanism**:Provide users with a way to provide feedback or report issues with the application.
6. **Responsive Design**: Ensure that the GUI is responsive and works well on different devices and screen sizes.
7. **Branding**: Incorporate branding elements such as logos, colors, and fonts to maintain consistency with your application or organization's branding.

4.4.3 Clustering and Classification

GLCM Algorithm

The Feature Extraction is a method of capturing visual content of images for indexing retrieval. Grey Level Co-occurrence Matrix (GLCM) method is a way of extracting second order statistical texture features. The GLCM functions characterize the texture of an image by calculating with specific values and in a specified spatial relationship occur in an image, creating a GLCM, and then extracting statistical measures from this matrix. The image segmentation is the process of partitioning a digital image into multiple segments (sets of pixels, also known as image objects). The goal of segmentation is to simplify and/or change the representation of an image into something that is more meaningful and easier to analyze.

K Means clustering

The k-means algorithm is a clustering algorithm that can be used for plant disease detection. The algorithm works by grouping similar data points into clusters, with each cluster being represented by its centroid. Initialize the system and input data which involves setting up the detection system and capturing images of the plants. K-means clustering is the extracted features are then used as input to the k-means algorithm. The algorithm groups the features into a specified number of clusters (k), with each cluster being represented by its centroid. The number of clusters is usually determined by trial and error, or by using techniques such as the elbow method.

Feature extraction:

The next step is to extract relevant features from the images. This could involve techniques such as color-based feature extraction or texture analysis.Classify images based on cluster labels, Once the clustering is complete, each image can be classified based on the cluster it belongs to. If the cluster contains images of healthy plants, then any new image that belongs to that cluster can be classified as healthy. Conversely, if the cluster contains images of diseased plants, then any new image that belongs to that cluster can be classified as diseased.

4.5 Steps to execute/run/implement the project

4.5.1 Data Collection

• **Patient Demographics**:Basic information about the patient, including age, gender, ethnicity, occupation, and lifestyle habits like smoking and alcohol consumption, which can help in understanding potential risk factors.
• **Medical History**: Past medical conditions and family history can provide insights into the patient's overall health and predisposition to certain diseases, including CKD. Conditions like hypertension, diabetes, and cardiovascular diseases are known risk factors for CKD.
• **Labeling Images**: Each image needs to be labeled with information such as the plant species, the disease present (if any), and the severity of the disease. This labeling process helps in training the machine learning model.
• **Imaging Studies**: Renal ultrasound and other imaging modalities can provide visual information about the size, shape, and structure of the kidneys, aiding in the diagnosis and monitoring of CKD.
• **Data Source**:Data can be sourced from electronic health records, laboratory databases, patient interviews, or remote monitoring devices, depending on availability and accessibility.
• **Data Preprocessing**: Before building predictive models, it's essential to clean and preprocess the data, handling missing values, outliers, and encoding categorical variables appropriately.

4.5.2 Model Selection

• **Understand the Data**:Begin by thoroughly understanding your dataset. Identify the features available for prediction and their relevance to CKD. Ensure the dataset is clean and properly preprocessed.
• **Define Performance Metrics**: Decide on the evaluation metrics based on the nature of your problem. For CKD prediction, common metrics include accuracy, precision, recall, F1-score, and area under the ROC curve (AUC-ROC).
• **Baseline Models**: Start with simple baseline models to establish a benchmark performance. These could include logistic regression, decision trees, or naive Bayes classifiers.
• **Regularization Techniques**: Consider using regularization techniques to prevent

overfitting in complex models. Common regularization methods include L1 and L2 regularization for linear models and dropout for neural networks.

• **Cross-Validation**: Employ cross-validation techniques such as k-fold cross-validation to assess the generalization performance of your models.

• **Model Evaluation**: Evaluate the performance of each model using the chosen metrics on both the training and validation datasets. Ensure that the selected model generalizes well to unseen data.

4.5.3 Model Evaluation and deployment

• **Data Collection and Preprocessing**: Collect relevant data on patients' demographics, medical history, laboratory tests, and other pertinent information related to CKD. Preprocess the data to handle missing values, outliers, and ensure consistency and quality.

• **Feature Selection and Engineering**: Identify the most relevant features for predicting CKD. Engineer new features if necessary to improve the predictive power of the model.

• **Model Training**:Select an appropriate machine learning algorithm (e.g., logistic regression, decision trees, random forests, support vector machines) or deep learning model.

• **Model Evaluation**: Evaluate the model's performance using appropriate metrics such as accuracy, precision, recall, F1-score, and area under the receiver operating characteristic curve (ROC-AUC).

• **Monitoring and Maintenance**: Implement monitoring systems to track the model's performance in real-time. Regularly update the model as new data becomes available or as the underlying patterns change.

• **Documentation**:Document the entire process, including data collection, preprocessing steps, model architecture, evaluation results, and deployment procedures.

Chapter 5

IMPLEMENTATION AND TESTING

5.1 Input and Output

5.1.1 Input Design

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	AA
1	id	age	bp	sg	al	su	rbc	pc	pcc	ba	bgr	bu	sc	sod	pot	hemo	pcv	wc	rc	htn	dm	cad	appet	pe	ane	classification	
2	0	48	80	1.02	1	0		normal	notpresen	notpresen	121	36	1.2			15.4	44	7800	5.2	yes	yes	no	good	no	no	ckd	
3	1	7	50	1.02	4	0		normal	notpresen	notpresent		18	0.8			11.3	38	6000		no	no	no	good	no	no	ckd	
4	2	62	80	1.01	2	3	normal	normal	notpresen	notpresen	423	53	1.8			9.6	31	7500		no	yes	no	poor	no	yes	ckd	
5	3	48	70	1.005	4	0	normal	abnormal	present	notpresen	117	56	3.8	111	2.5	11.2	32	6700	3.9	yes	no	no	poor	yes	yes	ckd	
6	4	51	80	1.01	2	0	normal	normal	notpresen	notpresen	106	26	1.4			11.6	35	7300	4.6	no	no	no	good	no	no	ckd	
7	5	60	90	1.015	3	0			notpresen	notpresen	74	25	1.1	142	3.2	12.2	39	7800	4.4	yes	yes	no	good	yes	no	ckd	
8	6	68	70	1.01	0	0		normal	notpresen	notpresen	100	54	24	104	4	12.4	36			no	no	no	good	no	no	ckd	
9	7	24		1.015	2	4	normal	abnormal	notpresen	notpresen	410	31	1.1			12.4	44	6900	5	no	yes	no	good	yes	no	ckd	
10	8	52	100	1.015	3	0	normal	abnormal	present	notpresen	138	60	1.9			10.8	33	9600	4	yes	yes	no	good	no	yes	ckd	
11	9	53	90	1.02	2	0	abnormal	abnormal	present	notpresen	70	107	7.2	114	3.7	9.5	29	12100	3.7	yes	yes	no	poor	no	yes	ckd	
12	10	50	60	1.01	2	4		abnormal	present	notpresen	490	55	4			9.4	28			yes	yes	no	good	no	yes	ckd	
13	11	63	70	1.01	3	0	abnormal	abnormal	present	notpresen	380	60	2.7	131	4.2	10.8	32	4500	3.8	yes	yes	no	poor	yes	no	ckd	
14	12	68	70	1.015	3	1		normal	present	notpresen	208	72	2.1	138	5.8	9.7	28	11200	3.4	yes	yes	yes	poor	yes	no	ckd	
15	13	68	70						notpresen	notpresen	98	86	4.6	135	3.4	9.8				yes	yes	yes	poor	yes	no	ckd	
16	14	68	80	1.01	3	2	normal	abnormal	present	present	157	90	4.1	130	6.4	5.6	16	11000	2.6	yes	yes	yes	poor	yes	no	ckd	
17	15	40	80	1.015	3	0		normal	notpresen	notpresen	76	162	9.8	141	4.9	7.6	24	3800	2.8	yes	no	no	good	no	yes	ckd	
18	16	47	70	1.015	2	0		normal	notpresen	notpresen	99	46	2.2	138	4.1	12.6				no	no	no	good	no	no	ckd	
19	17	47	80						notpresen	notpresen	114	87	5.2	139	3.7	12.1				yes	no	no	poor	no	no	ckd	
20	18	60	100	1.025	0	3		normal	notpresen	notpresen	263	27	1.3	135	4.3	12.7	37	11400	4.3	yes	yes	yes	good	no	no	ckd	
21	19	62	60	1.015	1	0		abnormal	present	notpresen	100	31	1.6			10.3	30	5300	3.7	yes	no	yes	good	no	no	ckd	
22	20	61	80	1.015	2	0	abnormal	abnormal	notpresen	notpresen	173	148	3.9	135	5.2	7.7	24	9200	3.2	yes	yes	yes	poor	yes	yes	ckd	
23	21	60	90						notpresen	notpresent		180	76	4.5		10.9	32	6300	3.6	yes	yes	yes	good	no	no	ckd	
24	22	48	80	1.025	4	0	normal	abnormal	notpresen	notpresen	95	163	7.7	136	3.8	9.8	32	6900	3.4	yes	no	no	good	no	yes	ckd	
25	23	21	70	1.01	0	0		normal	notpresen	notpresent										no	no	no	poor	no	yes	ckd	
26	24	42	100	1.015	4	0	normal	abnormal	notpresen	present		50	1.4	129	4	11.1	29	8300	4.8	yes	no	no	poor	no	no	ckd	
27	25	61	60	1.025	0	0		normal	notpresen	notpresen	108	75	1.9	141	5.2	9.9	29	8400	3.7	yes	yes	no	good	no	yes	ckd	
28	26	75	80	1.015	0	0		normal	notpresen	notpresen	156	45	2.4	140	3.4	11.6	35	10300	4	yes	yes	no	poor	no	no	ckd	
29	27	69	70	1.01	3	4	normal	abnormal	notpresen	notpresen	264	87	2.7	130	4	12.5	37	9600	4.1	yes	yes	yes	good	yes	no	ckd	
30	28	75	70		1	3			notpresen	notpresen	123	31	1.4							no	yes	no	good	no	no	ckd	
31	29	68	70	1.005	1	0	abnormal	abnormal	present	notpresent		28	1.4			12.8	38			no	no	yes	good	no	no	ckd	

Figure 5.1: **Input Data**

Figure 5.1 Here we are using the 2nd approach because this is an easy way and we can also increase the instances along with it. Currently, we are having 400 instances and a predictive model will work just fine with more instances.

5.1.2 Output Design

```
df.describe().T
```

	count	mean	std	min	25%	50%	75%	max
age	391.0	51.483376	17.169714	2.000	42.00	55.00	64.50	90.000
blood_pressure	388.0	76.469072	13.683637	50.000	70.00	80.00	80.00	180.000
specific_gravity	353.0	1.017408	0.005717	1.005	1.01	1.02	1.02	1.025
albumin	354.0	1.016949	1.352679	0.000	0.00	0.00	2.00	5.000
sugar	351.0	0.450142	1.099191	0.000	0.00	0.00	0.00	5.000
blood_glucose_random	356.0	148.036517	79.281714	22.000	99.00	121.00	163.00	490.000
blood_urea	381.0	57.425722	50.503006	1.500	27.00	42.00	66.00	391.000
serum_creatinine	383.0	3.072454	5.741126	0.400	0.90	1.30	2.80	76.000
sodium	313.0	137.528754	10.408752	4.500	135.00	138.00	142.00	163.000
potassium	312.0	4.627244	3.193904	2.500	3.80	4.40	4.90	47.000
haemoglobin	348.0	12.526437	2.912587	3.100	10.30	12.65	15.00	17.800

Figure 5.2: **Output Data**

Figure 5.2 The output of this predictive is represented numerically ranging between 0 to 1 because we are using logistic regression here and this initiative gives the output in the range 0 and 1. And this is common for every model, the output will be in the range of 0 and 1.

5.2 Testing

A strategy in this testing is integrating the systems test cases along with the design techniques into a well planned series of steps with the results in the successful construction of software. This testing strategy should work along with the test planning, test execution, test case design, and the data collection resultant and evaluation. To accommodate a low-level test is a strategy for software testing. Software testing is a critical element of software quality assurance which represents the ultimate review of specification design and coding. Interesting anomalies from the software are being to represented by using Testing. Thus, the period of testing is performed for the proposed system and there by is ready for user acceptance testing.

5.3 Types of Testing

5.3.1 Unit testing

Chronic kidney disease is a medical condition characterized by the gradual loss of kidney function over time. It can have various causes and stages, and its management often involves medical interventions, lifestyle changes, and sometimes software tools for tracking and managing patient data.

Input

```
models = {
    "Random Forest": RandomForestClassifier(),

}

for model_name, model in models.items():
    model.fit(X_train, y_train)

    # Make predictions on the test set
    y_pred = model.predict(X_test)

    # Evaluate the model
    accuracy = accuracy_score(y_test, y_pred)
    print(f'{model_name} Accuracy: {accuracy:.2f}')

    # Generate a classification report
    print(f'{model_name} Classification Report:')
    print(classification_report(y_test, y_pred))

    cm = confusion_matrix(y_test, y_pred)
    print(f'{model_name} Confusion Matrix:')
    print(cm)

    print('-' * 55)
```

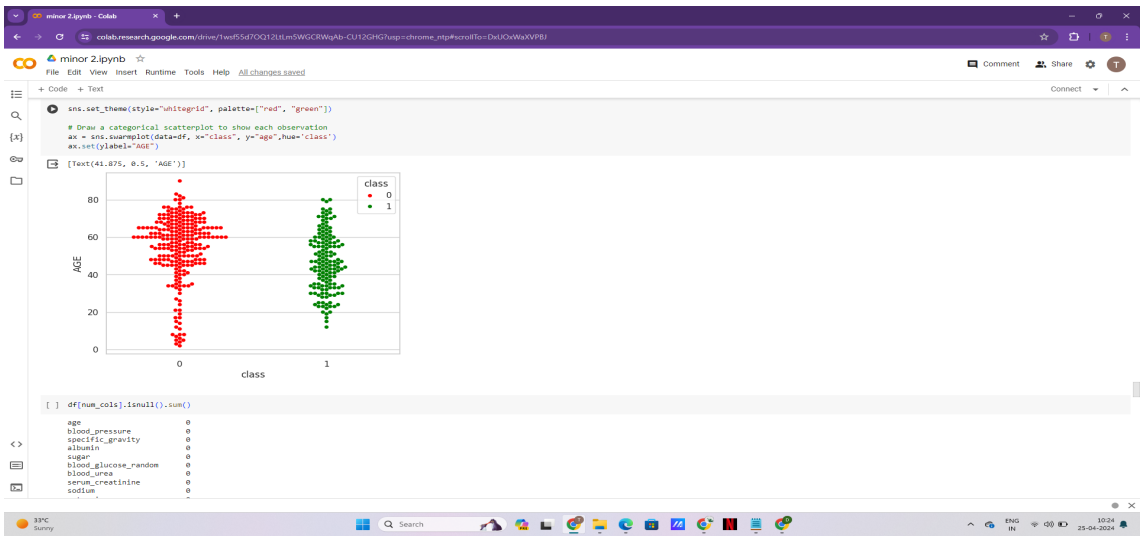


Figure 5.3: **Test result of Unit Testing**

Test result : PASSED

5.3.2 Integration testing

Integration testing for chronic kidney disease (CKD) involves examining how different components of CKD management interact to ensure the entire system works effectively. CKD management typically involves various healthcare professionals, medical interventions, medications, and lifestyle modifications.

Input

```
df.sample(10)
df.info()
df.describe().T
df['packed_cell_volume'] = pd.to_numeric(df['packed_cell_volume'], errors='coerce')
df['white_blood_cell_count'] = pd.to_numeric(df['white_blood_cell_count'], errors='coerce')
df['red_blood_cell_count'] = pd.to_numeric(df['red_blood_cell_count'], errors='coerce')
cat_cols = [col for col in df.columns if df[col].dtype == 'object']
cat_cols
num_cols = [col for col in df.columns if df[col].dtype != 'object']
num_cols
for col in cat_cols:
    print(f'{col} {df[col].unique()}')
df['diabetes_mellitus'].replace({'\tno':'no', '\tyes':'yes', ' yes':'yes'}, inplace=True)
df['coronary_artery_disease'] = df['coronary_artery_disease'].str.replace('\tno', 'no')
df['class'].replace({'ckd\t': 'ckd', 'notckd': 'not ckd'}, inplace=True)
df['class'] = df['class'].map({'ckd': 0, 'not ckd': 1})
df['class'] = pd.to_numeric(df['class'], errors='coerce')
cols = ['diabetes_mellitus','coronary_artery_disease','class']
```

```
for col in cols:
    print(f'{col} {df[col].unique()}')
value_counts = df['class'].value_counts()
categories = value_counts.index
counts = value_counts.values
colors = ['lightgreen', 'seagreen', 'forestgreen']
explode = [0.05] * len(counts)
```

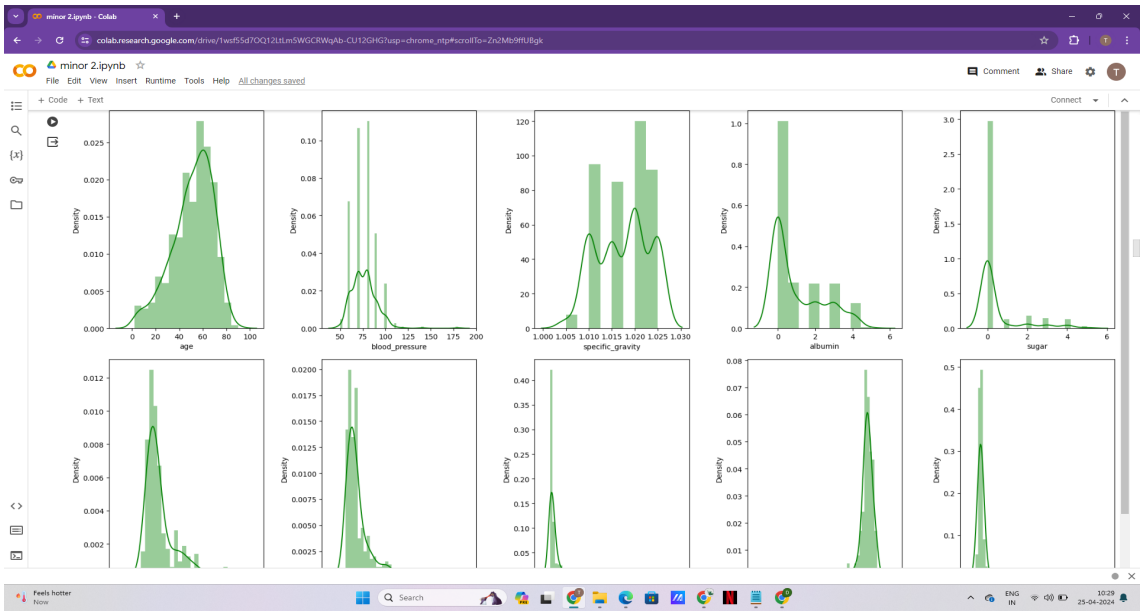


Figure 5.4: **Integration testing result**

Test result : PASSED

5.3.3 System testing

A strategy in this system testing is integrating the systems test cases along with the design techniques into a well planned series of steps with the results in the successful construction of software. This testing strategy should work along with the test planning, test execution, test case design, and the data collection resultant and evaluation. To accommodate a low-level test is a strategy for software testing. Software testing is a critical element of software quality assurance which represents the ultimate review of specification design and coding. Interesting anomalies from the software are being to represented by using Testing. Thus, the period of testing is performed for the proposed system and there by is ready for user acceptance testing.

Input

```
plt.title('Blood Pressure Density')
plt.xlabel('Blood Pressure')
plt.ylabel('Density')
plt.legend()
plt.show()
normal_wbc = df[(df['white_blood_cell_count'] >= 4500) & (df['white_blood_cell_count'] <= 11000)]['white_blood_cell_count']
low_wbc = df[df['white_blood_cell_count'] < 4500]['white_blood_cell_count']
high_wbc = df[df['white_blood_cell_count'] > 11000]['white_blood_cell_count']

plt.figure(figsize=(12, 6))
sns.kdeplot(data=normal_wbc, color='green', shade=True, label='Normal White Blood Cell Count (4500 - 11000)')
sns.kdeplot(data=low_wbc, color='blue', shade=True, label='Low White Blood Cell Count (< 4500)')
sns.kdeplot(data=high_wbc, color='red', shade=True, label='High White Blood Cell Count (> 11000)')

plt.title('White Blood Cell Count Density')
plt.xlabel('White Blood Cell Count')
plt.ylabel('Density')
plt.legend()
plt.show()
normal_rbc = df[((df['red_blood_cell_count'] >= 4.35) & (df['red_blood_cell_count'] <= 5.65)) | ((df['red_blood_cell_count'] >= 3.92) & (df['red_blood_cell_count'] <= 5.13))]['red_blood_cell_count']
low_rbc = df[(df['red_blood_cell_count'] < 4.35) | (df['red_blood_cell_count'] < 3.92)]['red_blood_cell_count']
high_rbc = df[(df['red_blood_cell_count'] > 5.65) | (df['red_blood_cell_count'] > 5.13)]['red_blood_cell_count']

plt.figure(figsize=(12, 6))
sns.kdeplot(data=normal_rbc, color='green', shade=True, label='Normal Red Blood Cell Count (4.35 - 5.65)')
sns.kdeplot(data=low_rbc, color='blue', shade=True, label='Low Red Blood Cell Count (< 4.35 for men, < 3.92 for women)')
sns.kdeplot(data=high_rbc, color='red', shade=True, label='High Red Blood Cell Count (> 5.65 for men, > 5.13 for women)')

plt.title('Red Blood Cell Count Density')
plt.xlabel('Red Blood Cell Count')
plt.ylabel('Density')
plt.legend()
plt.show()
normal_hb_male = df[(df['haemoglobin'] >= 14) & (df['haemoglobin'] <= 18)]['haemoglobin']
normal_hb_female = df[(df['haemoglobin'] >= 12) & (df['haemoglobin'] <= 16)]['haemoglobin']
low_hb = df[(df['haemoglobin'] < 12) | (df['haemoglobin'] < 14)]['haemoglobin']
high_hb = df[(df['haemoglobin'] > 16) | (df['haemoglobin'] > 18)]['haemoglobin']
```

Test Result : PASSED

5.3.4 Test Result

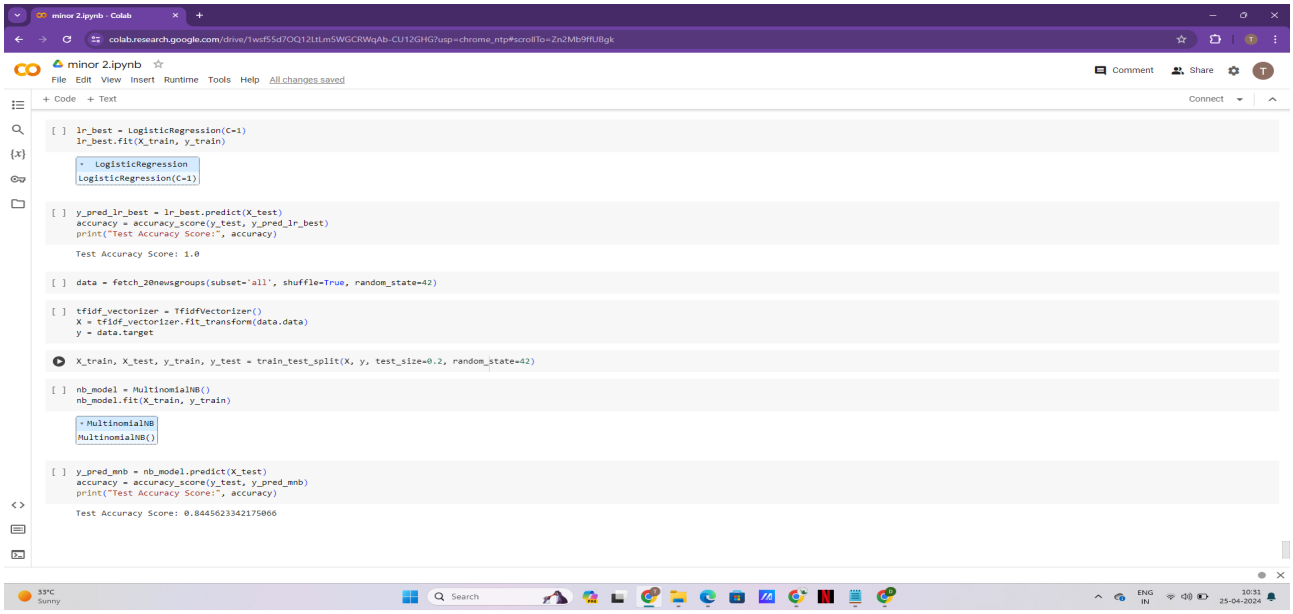


Figure 5.5: **Test Image**

Chapter 6

RESULTS AND DISCUSSIONS

6.1 Efficiency of the Proposed System

As we have mentioned the earlier that it is better to search for an alternative approach with promising results and we are going to do that. So, in this study, we are going to compare four very commonly known machine learning algorithms and we are going to state the best among them using a case study on Chronic Kidney disease Prediction. As stated, the four machine learning algorithms are: Random Forest Decision Tree LGBM K- Near Neighbour (KNN) Why we are going with only four algorithms is that we don't want to overload and confuse the learning procedure by including more machine learning algorithms. So, we thought that four would be an optimal choice and the machine would work just fine. Using confusion matrix we received the accuracy of all four methodologies such as K-Nearest Neighbor, Random Forest, Decision Tree and Light Gradient Boosted Machine. We receive the accuracy of KNN is 94 percentage and Random Forest, Decision Tree, Light Gradient Boosted Machine accuracy are 98 percentage. Compared to Light Gradient Boosted Machine, Decision Tree and Random Forest we receive less accuracy in K-Nearest Neighbor.

6.2 Comparison of Existing and Proposed System

```
Random Forest Accuracy: 1.00
Random Forest Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        52
           1       1.00      1.00      1.00        28

    accuracy                           1.00        80
   macro avg       1.00      1.00      1.00        80
weighted avg       1.00      1.00      1.00        80

Random Forest Confusion Matrix:
[[52  0]
 [ 0 28]]
```

Figure 6.1: **Random Forest accuracy**

```
Logistic Regression Accuracy: 0.93
Logistic Regression Classification Report:
              precision    recall  f1-score   support

           0       0.94      0.94      0.94        52
           1       0.89      0.89      0.89        28

    accuracy                           0.93        80
   macro avg       0.92      0.92      0.92        80
weighted avg       0.93      0.93      0.93        80

Logistic Regression Confusion Matrix:
[[49  3]
 [ 3 25]]
```

Figure 6.2: **Logistic Regression accuracy**

6.3 Sample Code

```python
# Importing libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, MinMaxScaler
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from keras.models import Sequential
from keras.layers import Dense
from keras.initializers import RandomNormal

# Read CSV file
df = pd.read_csv('chronickidney.csv')

# Selecting columns to retain
columns_to_retain = ['sg', 'al', 'sc', 'hemo', 'pcv', 'wbcc', 'rbcc', 'htn', 'classification']
df = df[columns_to_retain].dropna()

# Encoding categorical variables
for column in df.columns:
    if df[column].dtype != np.number:
        df[column] = LabelEncoder().fit_transform(df[column])

# Scaling features
x = df.drop('classification', axis=1)
y = df['classification']
scaler = MinMaxScaler()
x_scaled = scaler.fit_transform(x)

# Splitting data into training and testing sets
x_train, x_test, y_train, y_test = train_test_split(x_scaled, y, test_size=0.2, shuffle=True)

# Building a neural network model
model = Sequential()
model.add(Dense(256, input_dim=len(x.columns), kernel_initializer=RandomNormal(seed=13), activation=
    'relu'))
model.add(Dense(1, activation='hard_sigmoid'))
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])

# Training the model
history = model.fit(x_train, y_train, epochs=399, batch_size=x_train.shape[0])

# Saving the model
model.save('ckd.model')
```

```python

# Plotting accuracy and loss
plt.plot(history.history['accuracy'])
plt.plot(history.history['loss'])
plt.title('Model Accuracy & Loss')
plt.ylabel('Accuracy and Loss')
plt.xlabel('Epoch')
plt.legend(['Accuracy', 'Loss'], loc='upper right')
plt.show()

# Printing shapes of training and testing data
print('Shape of training data:', x_train.shape)
print('Shape of testing data:', x_test.shape)

# Making predictions
pred = model.predict(x_test)
pred = [1 if y >= 0.5 else 0 for y in pred]

# Printing original and predicted values
print('Original:', ", ".join(str(x) for x in y_test))
print('Predicted:', ", ".join(str(x) for x in pred))

# KNN Classifier
knn = KNeighborsClassifier()
knn.fit(x_train, y_train)
knn_acc = accuracy_score(y_test, knn.predict(x_test))
print(f"Training Accuracy of KNN is {accuracy_score(y_train, knn.predict(x_train))}")
print(f"Test Accuracy of KNN is {knn_acc}")
print(f"Confusion Matrix:\n{confusion_matrix(y_test, knn.predict(x_test))}\n")
print(f"Classification Report:\n{classification_report(y_test, knn.predict(x_test))}")

# Decision Tree Classifier
dtc = DecisionTreeClassifier()
dtc.fit(x_train, y_train)
dtc_acc = accuracy_score(y_test, dtc.predict(x_test))
print(f"Training Accuracy of Decision Tree Classifier is {accuracy_score(y_train, dtc.predict(x_train))}")
print(f"Test Accuracy of Decision Tree Classifier is {dtc_acc}")
print(f"Confusion Matrix:\n{confusion_matrix(y_test, dtc.predict(x_test))}\n")
print(f"Classification Report:\n{classification_report(y_test, dtc.predict(x_test))}")

# Random Forest Classifier
rdclf = RandomForestClassifier(criterion='entropy', max_depth=11, max_features='auto', min_samples_leaf=2,
                               min_samples_split=3, n_estimators=130)
rdclf.fit(x_train, y_train)
rdclf_acc = accuracy_score(y_test, rdclf.predict(x_test))
print(f"Training Accuracy of Random Forest Classifier is {accuracy_score(y_train, rdclf.predict(x_train))}")
print(f"Test Accuracy of Random Forest Classifier is {rdclf_acc}")
```

```
print(f"Confusion Matrix:\n{confusion_matrix(y_test, rdclf.predict(x_test))}\n")
print(f"Classification Report:\n{classification_report(y_test, rdclf.predict(x_test))}")

# Sorting models based on accuracy
models = pd.DataFrame({
    'Model': ['KNN', 'Decision Tree Classifier', 'Random Forest Classifier'],
    'Score': [knn_acc, dtc_acc, rdclf_acc]
})
models = models.sort_values(by='Score', ascending=False)

print(models)
```

Output

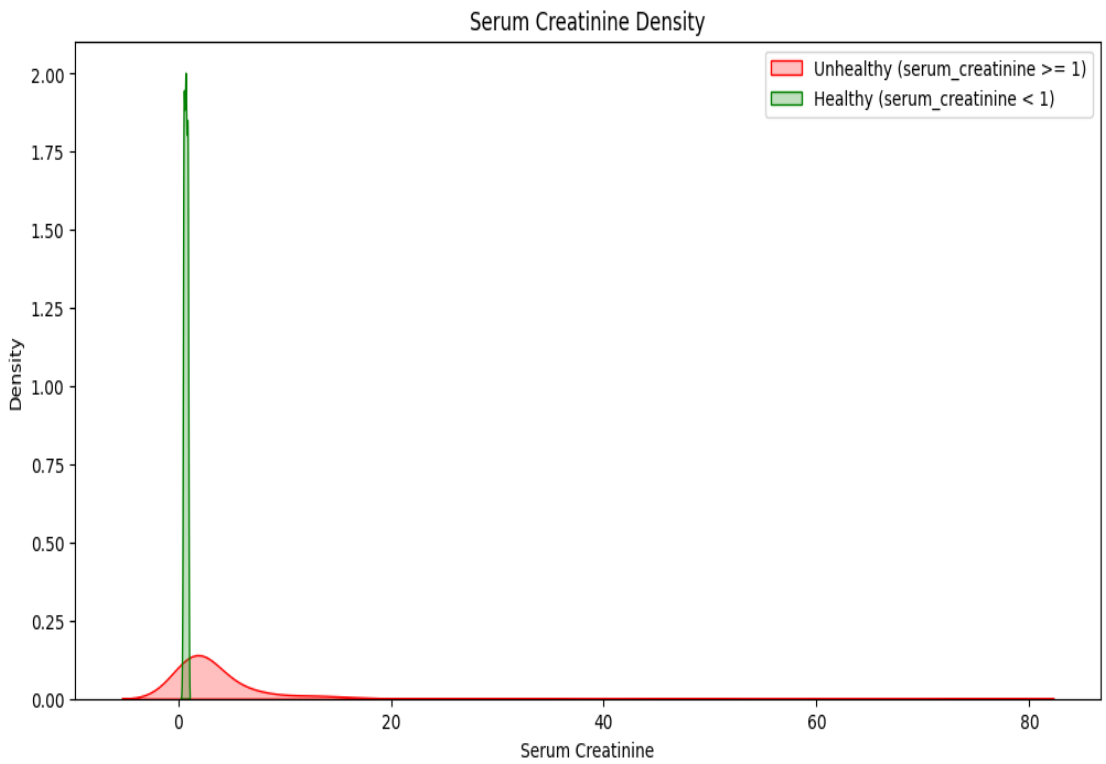


Figure 6.3: **Serum Creatinine Density**

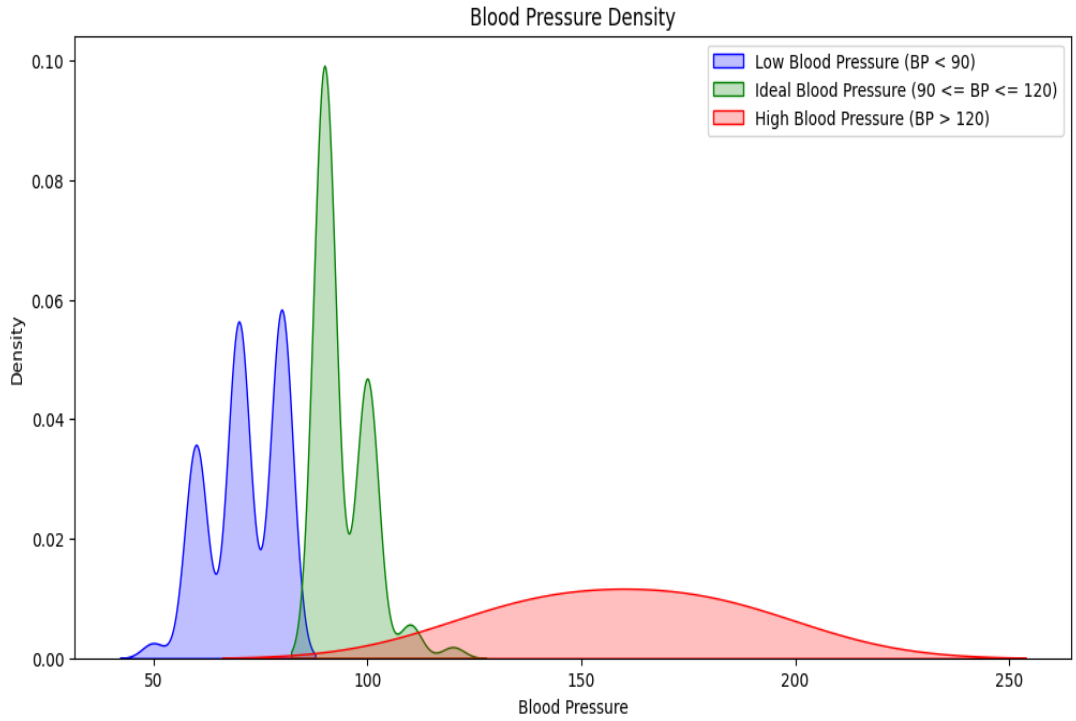


Figure 6.4: **Blood pressure Density**

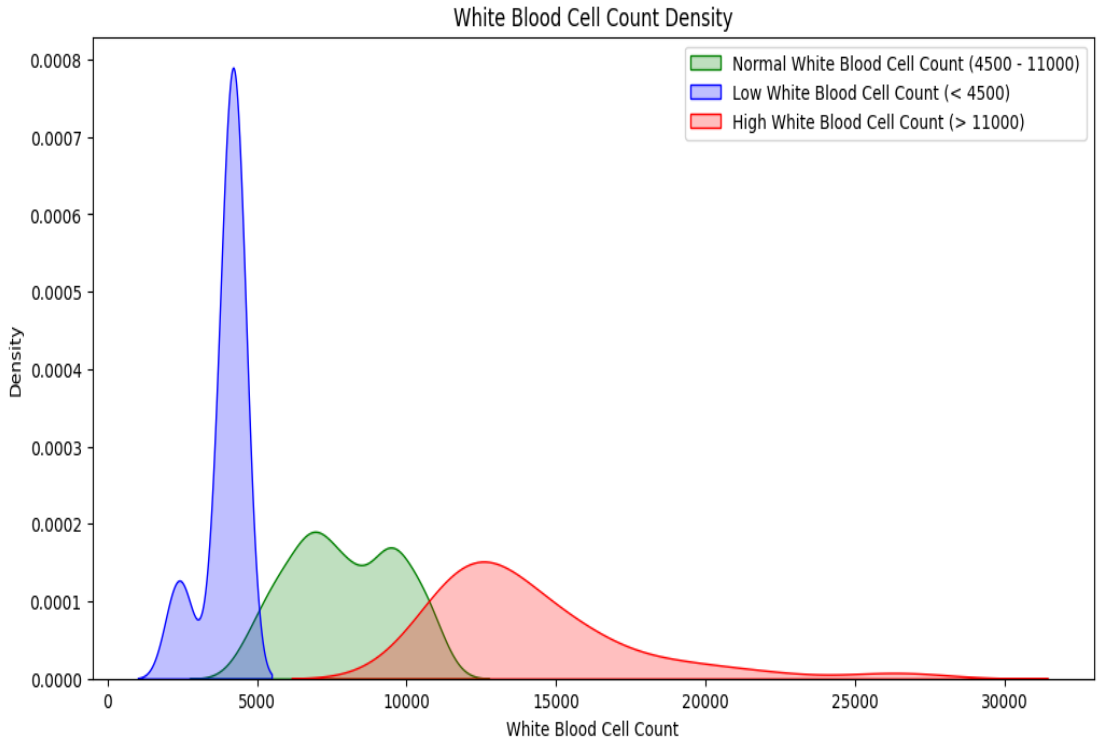


Figure 6.5: **White Blood Cell Count Density**

Chapter 7

CONCLUSION AND FUTURE ENHANCEMENTS

7.1 Conclusion

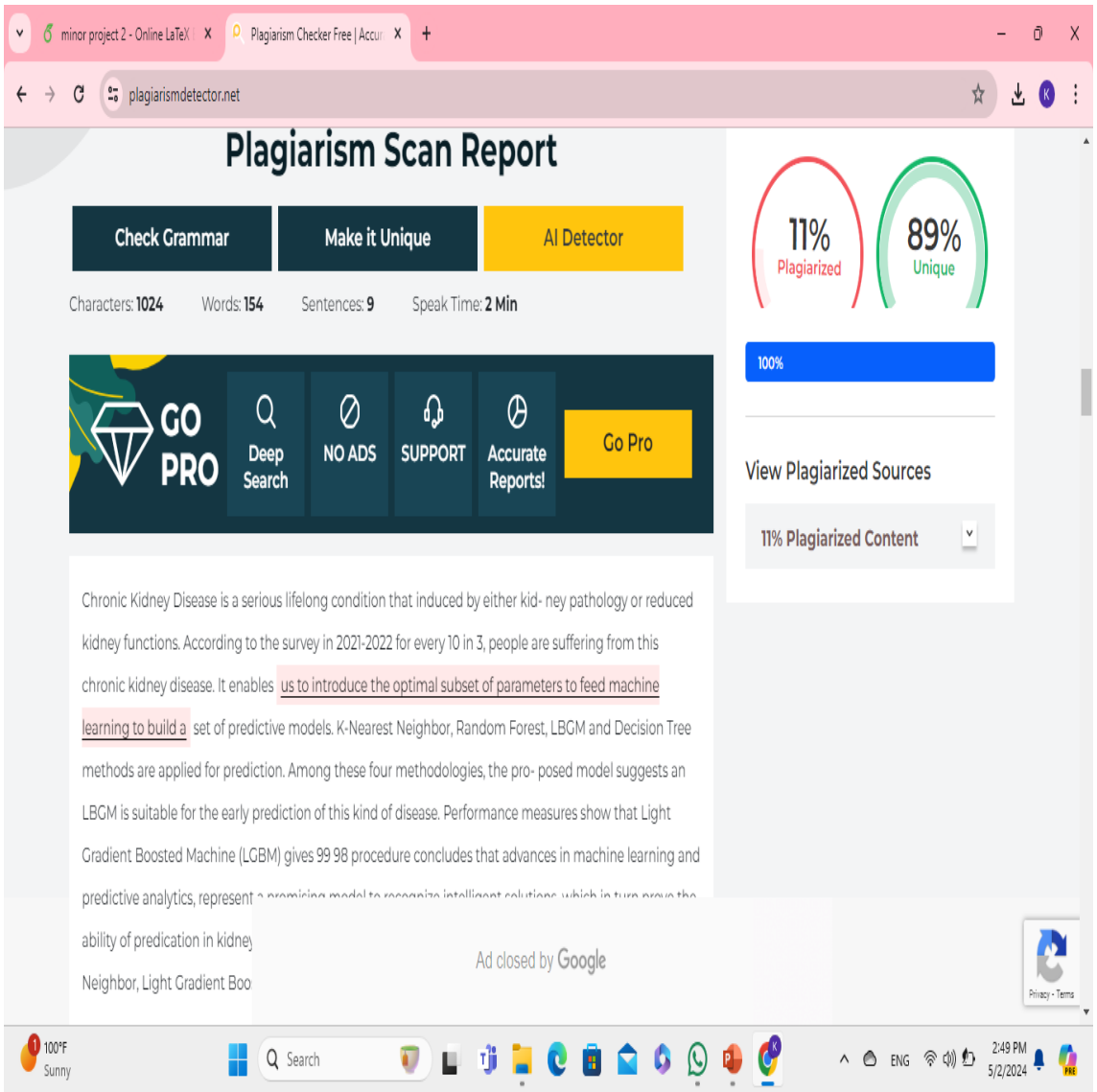
CKD means that the kidney does not work and cannot correctly filter the blood. About 10 percent of the population for a worldwide suffering from (CKD), and millions of die each year because of they couldn't get the affordable treatment, with the number in- creasing in the elderly. In 2010, a study was conducted by International Society of Numerology(ISN) on global burden disease, they reported that CKD has been raised an important cause of the mortality worldwide with the number of deaths increasing by 82.3 percent in the last two decades. Also, the number of patients reaching End-Stage Renal Disease (ESRD) is increasing, which requires kidney transplantation or dialysis to save the patient‘s lives.

7.2 Future Enhancements

As a complementary solution, we could build a mobile application, to make it available to everyone to detect whether they are CKD or not and as of now if we have a correct data set we could use it to predict many more diseases and also if the application collaborates with a highly reputed company then we can allow making more advancements to the existing model. There would be disadvantages because of we need to have higher graphic card compatibility to run the model and the solution for that would be, the involvement of a framework such as Kara's or Tensor- Flow at the back end.

Chapter 8

PLAGIARISM REPORT



Chapter 9

SOURCE CODE & POSTER PRESENTATION

9.1 Source Code

```

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
from sklearn.preprocessing import OneHotEncoder
from sklearn.compose import ColumnTransformer

from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier, AdaBoostClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import MultinomialNB
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, classification_report,confusion_matrix,precision_score,
    f1_score, recall_score
from sklearn.metrics import roc_curve, roc_auc_score
from sklearn.metrics import precision_recall_curve, auc
from sklearn.model_selection import GridSearchCV
from sklearn.datasets import load_iris
from sklearn.datasets import fetch_20newsgroups
from sklearn.feature_extraction.text import TfidfVectorizer


import warnings
warnings.filterwarnings('ignore')
%matplotlib inline
pd.set_option('display.max_columns', 26)
df = pd.read_csv('kidney_disease.csv')
df.head()
df.drop('id', axis=1, inplace=True)
df.head()
```

```
df.columns = ['age', 'blood_pressure', 'specific_gravity', 'albumin', 'sugar', 'red_blood_cells', 'pus_cell',
              'pus_cell_clumps', 'bacteria', 'blood_glucose_random', 'blood_urea', 'serum_creatinine', 'sodium',
              'potassium', 'haemoglobin', 'packed_cell_volume', 'white_blood_cell_count', 'red_blood_cell_count',
              'hypertension', 'diabetes_mellitus', 'coronary_artery_disease', 'appetite', 'peda_edema',
              'aanemia', 'class']
df.sample(10)
df.info()
df.describe().T
df['packed_cell_volume'] = pd.to_numeric(df['packed_cell_volume'], errors='coerce')
df['white_blood_cell_count'] = pd.to_numeric(df['white_blood_cell_count'], errors='coerce')
df['red_blood_cell_count'] = pd.to_numeric(df['red_blood_cell_count'], errors='coerce')
cat_cols = [col for col in df.columns if df[col].dtype == 'object']
cat_cols
num_cols = [col for col in df.columns if df[col].dtype != 'object']
num_cols
for col in cat_cols:
    print(f'{col} {df[col].unique()}')
df['diabetes_mellitus'].replace({'\tno':'no', '\tyes':'yes', ' yes':'yes'}, inplace=True)
df['coronary_artery_disease'] = df['coronary_artery_disease'].str.replace('\tno', 'no')
df['class'].replace({'ckd\t': 'ckd', 'notckd': 'not ckd'}, inplace=True)
df['class'] = df['class'].map({'ckd': 0, 'not ckd': 1})
df['class'] = pd.to_numeric(df['class'], errors='coerce')
cols = ['diabetes_mellitus','coronary_artery_disease','class']

for col in cols:
    print(f'{col} {df[col].unique()}')
value_counts = df['class'].value_counts()
categories = value_counts.index
counts = value_counts.values
colors = ['lightgreen', 'seagreen', 'forestgreen']
explode = [0.05] * len(counts)

plt.pie(counts, explode=explode, shadow=True, labels=categories, autopct='%1.1f%%', startangle=140, pctdistance=0.85, colors=colors)
plt.axis('equal')
plt.title('Value Count')
plt.legend(categories, title='Classes', loc='best')
plt.show()
sns.pairplot(df);
plt.figure(figsize=(20, 15))
for i, column in enumerate(num_cols, 1):
    plt.subplot(3, 5, i)
    sns.distplot(df[column], color='green')
    plt.xlabel(column)

plt.tight_layout()
```

```
plt.show()
plt.figure(figsize=(20, 15))
for i, column in enumerate(cat_cols, 1):
    plt.subplot(3, 4, i)
    sns.countplot(data=df, x=column, palette='husl')
    plt.xlabel(column)
    plt.ylabel('Count')

plt.tight_layout()
plt.show()
for column in cat_cols:
    print(f"--- {column} ---")
    total_count = len(df[column])
    unique_values = df[column].value_counts()
    for value, count in unique_values.items():
        percentage = (count / total_count) * 100
        print(f"{value}: {percentage:.1f}%")
    print()
unhealthy = df[df['serum_creatinine'] >= 1]['serum_creatinine']
healthy = df[df['serum_creatinine'] < 1]['serum_creatinine']

plt.figure(figsize = (12,6))
sns.kdeplot(data=unhealthy, color='red', shade=True, label='Unhealthy (serum_creatinine >= 1)')
sns.kdeplot(data=healthy, color='green', shade=True, label='Healthy (serum_creatinine < 1)')

plt.title('Serum Creatinine Density')
plt.xlabel('Serum Creatinine')
plt.ylabel('Density')
plt.legend()
plt.show()
low_bp = df[df['blood_pressure'] < 90]['blood_pressure']
ideal_bp = df[(df['blood_pressure'] >= 90) & (df['blood_pressure'] <= 120)]['blood_pressure']
high_bp = df[df['blood_pressure'] > 120]['blood_pressure']

plt.figure(figsize=(12, 6))
sns.kdeplot(data=low_bp, color='blue', shade=True, label='Low Blood Pressure (BP < 90)')
sns.kdeplot(data=ideal_bp, color='green', shade=True, label='Ideal Blood Pressure (90 <= BP <= 120)'
    )
sns.kdeplot(data=high_bp, color='red', shade=True, label='High Blood Pressure (BP > 120)')

plt.title('Blood Pressure Density')
plt.xlabel('Blood Pressure')
plt.ylabel('Density')
plt.legend()
plt.show()
normal_wbc = df[(df['white_blood_cell_count'] >= 4500) & (df['white_blood_cell_count'] <= 11000)]['
    white_blood_cell_count']
low_wbc = df[df['white_blood_cell_count'] < 4500]['white_blood_cell_count']
high_wbc = df[df['white_blood_cell_count'] > 11000]['white_blood_cell_count']

```

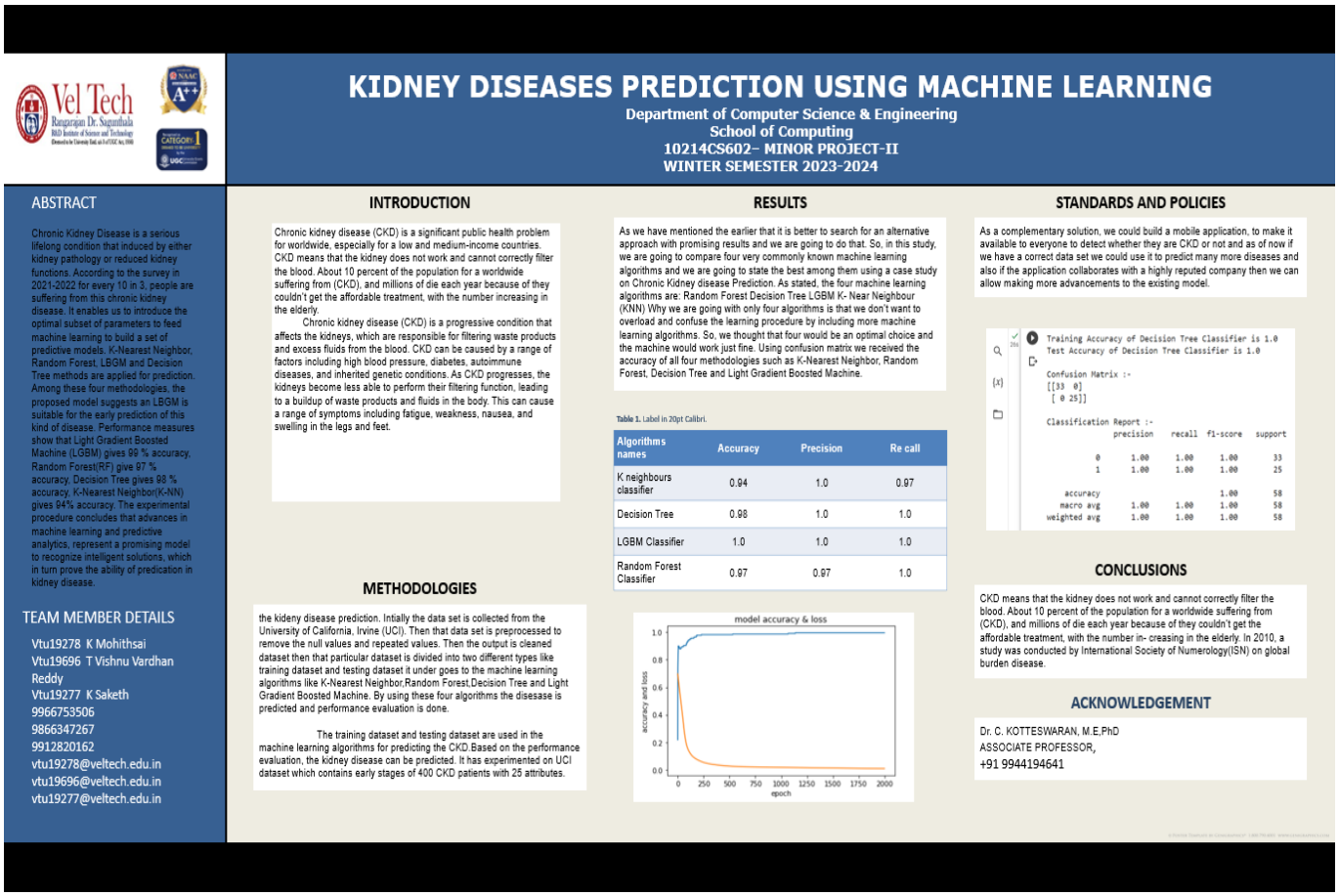
```
plt.figure(figsize=(12, 6))
sns.kdeplot(data=normal_wbc, color='green', shade=True, label='Normal White Blood Cell Count (4500 - 11000)')
sns.kdeplot(data=low_wbc, color='blue', shade=True, label='Low White Blood Cell Count (< 4500)')
sns.kdeplot(data=high_wbc, color='red', shade=True, label='High White Blood Cell Count (> 11000)')

plt.title('White Blood Cell Count Density')
plt.xlabel('White Blood Cell Count')
plt.ylabel('Density')
plt.legend()
plt.show()
normal_rbc = df[((df['red_blood_cell_count'] >= 4.35) & (df['red_blood_cell_count'] <= 5.65)) | ((df['red_blood_cell_count'] >= 3.92) & (df['red_blood_cell_count'] <= 5.13))]['red_blood_cell_count']
low_rbc = df[(df['red_blood_cell_count'] < 4.35) | (df['red_blood_cell_count'] < 3.92)]['red_blood_cell_count']
high_rbc = df[(df['red_blood_cell_count'] > 5.65) | (df['red_blood_cell_count'] > 5.13)]['red_blood_cell_count']

plt.figure(figsize=(12, 6))
sns.kdeplot(data=normal_rbc, color='green', shade=True, label='Normal Red Blood Cell Count (4.35 - 5.65)')
sns.kdeplot(data=low_rbc, color='blue', shade=True, label='Low Red Blood Cell Count (< 4.35 for men, < 3.92 for women)')
sns.kdeplot(data=high_rbc, color='red', shade=True, label='High Red Blood Cell Count (> 5.65 for men, > 5.13 for women)')

plt.title('Red Blood Cell Count Density')
plt.xlabel('Red Blood Cell Count')
plt.ylabel('Density')
plt.legend()
plt.show()
normal_hb_male = df[(df['haemoglobin'] >= 14) & (df['haemoglobin'] <= 18)]['haemoglobin']
normal_hb_female = df[(df['haemoglobin'] >= 12) & (df['haemoglobin'] <= 16)]['haemoglobin']
low_hb = df[(df['haemoglobin'] < 12) | (df['haemoglobin'] < 14)]['haemoglobin']
high_hb = df[(df['haemoglobin'] > 16) | (df['haemoglobin'] > 18)]['haemoglobin']

# KDE grafi ini  izme
plt.figure(figsize=(12, 6))
sns.kdeplot(data=normal_hb_male, color='green', shade=True, label='Normal Haemoglobin (14 - 18) for males')
sns.kdeplot(data=normal_hb_female, color='blue', shade=True, label='Normal haemoglobin (12 - 16) for females')
sns.kdeplot(data=low_hb, color='red', shade=True, label='Low Haemoglobin (< 12 for both)')
sns.kdeplot(data=high_hb, color='purple', shade=True, label='High Haemoglobin (> 16 for females, > 18 for males)')

plt.title('Hemoglobin Level Density')
plt.xlabel('Hemoglobin Level (g/dl)')
plt.ylabel('Density')
```

```
plt.legend()
plt.show()
normal_sodium = df[(df['sodium'] >= 135) & (df['sodium'] <= 145)]['sodium']
low_sodium = df[df['sodium'] < 135]['sodium']
high_sodium = df[df['sodium'] > 145]['sodium']

plt.figure(figsize=(12, 6))
sns.kdeplot(data=normal_sodium, color='green', shade=True, label='Normal Sodium (135 - 145)')
sns.kdeplot(data=low_sodium, color='blue', shade=True, label='Low Sodium (< 135)')
sns.kdeplot(data=high_sodium, color='red', shade=True, label='High Sodium (> 145)')

plt.title('Sodium Level Density')
plt.xlabel('Sodium Level (mEq/L)')
plt.ylabel('Density')
plt.legend()
plt.show()
normal_urea = df[(df['blood_urea'] >= 5) & (df['blood_urea'] <= 20)]['blood_urea']
low_urea = df[df['blood_urea'] < 5]['blood_urea']
high_urea = df[df['blood_urea'] > 20]['blood_urea']

plt.figure(figsize=(12, 6))
sns.kdeplot(data=normal_urea, color='green', shade=True, label='Normal Urea (5 - 20)')
sns.kdeplot(data=low_urea, color='blue', shade=True, label='Low Urea (< 5)')
sns.kdeplot(data=high_urea, color='red', shade=True, label='High Urea (> 20)')

plt.title('Blood Urea Level Density')
plt.xlabel('Blood Urea Level (mg/dl)')
plt.ylabel('Density')
plt.legend()
plt.show()
sns.set_theme(style="whitegrid", palette=["red", "green"])

# Draw a categorical scatterplot to show each observation
ax = sns.swarmplot(data=df, x="class", y="age",hue='class')
ax.set(ylabel="AGE")
df[num_cols].isnull().sum()
median_values = df[num_cols].median()
df[num_cols] = df[num_cols].fillna(median_values)
df[num_cols].isnull().sum()
df[cat_cols].isnull().sum()
df[cat_cols].isnull().sum()
df[cat_cols].isnull().sum()
LabelEncoder = LabelEncoder()

for col in cat_cols:
    df[col] = LabelEncoder.fit_transform(df[col])
df.head()
y = df['class']
X = df.drop(['class'], axis=1)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 42)
```

```
models = {
    "Random Forest": RandomForestClassifier(),
    "Logistic Regression": LogisticRegression(),
    "K-Nearest Neighbors": KNeighborsClassifier(),
    "MultinomialNB": MultinomialNB(),
    "Decision Tree": DecisionTreeClassifier(),
    "AdaBoost": AdaBoostClassifier(),
}

for model_name, model in models.items():
    model.fit(X_train, y_train)

    # Make predictions on the test set
    y_pred = model.predict(X_test)

    # Evaluate the model
    accuracy = accuracy_score(y_test, y_pred)
    print(f'{model_name} Accuracy: {accuracy:.2f}')

    # Generate a classification report
    print(f'{model_name} Classification Report:')
    print(classification_report(y_test, y_pred))

    cm = confusion_matrix(y_test, y_pred)
    print(f'{model_name} Confusion Matrix:')
    print(cm)

    print('-' * 55)
lr = LogisticRegression()
lr.fit(X_train, y_train)
y_pred_lr = lr.predict(X_test)
print('Accuracy','%.3f' % accuracy_score(y_test, y_pred_lr))
print('Precision','%.3f' % precision_score(y_test, y_pred_lr))
print('Recall','%.3f' % recall_score(y_test, y_pred_lr))
print('F1 Score','%.3f' % f1_score(y_test, y_pred_lr))
iris = load_iris()
X = iris.data
y = iris.target
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
lr = LogisticRegression()
param_grid = {'C': [0.001, 0.01, 0.1, 1, 10, 100, 1000]}
grid_search = GridSearchCV(lr, param_grid, cv=5, scoring='accuracy')
grid_search.fit(X_train, y_train)

# En iyi hiperparametreleri ve do ruluk skorunu yazd rma
print("Best Params:", grid_search.best_params_)
print("Best accuracy score:", grid_search.best_score_)
lr_best = LogisticRegression(C=1)
lr_best.fit(X_train, y_train)
y_pred_lr_best = lr_best.predict(X_test)
```

```
accuracy = accuracy_score(y_test, y_pred_lr_best)
print("Test Accuracy Score:", accuracy)
data = fetch_20newsgroups(subset='all', shuffle=True, random_state=42)
tfidf_vectorizer = TfidfVectorizer()
X = tfidf_vectorizer.fit_transform(data.data)
y = data.target
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
nb_model = MultinomialNB()
nb_model.fit(X_train, y_train)
y_pred_mnb = nb_model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred_mnb)
print("Test Accuracy Score:", accuracy)
```

9.2 Poster Presentation



KIDNEY DISEASES PREDICTION USING MACHINE LEARNING

Department of Computer Science & Engineering
School of Computing
10214CS602– MINOR PROJECT-II
WINTER SEMESTER 2023-2024

ABSTRACT

Chronic Kidney Disease is a serious lifelong condition that induced by either kidney pathology or reduced kidney functions. According to the survey in 2021-2022 for every 10 in 3, people are suffering from this chronic kidney disease. It enables us to introduce the optimal subset of parameters to feed machine learning to build a set of predictive models. K-Nearest Neighbor, Random Forest, LBGM and Decision Tree methods are applied for prediction. Among these four methodologies, the proposed model suggests an LBGM is suitable for the early prediction of this kind of disease. Performance measures show that Light Gradient Boosted Machine (LGBM) gives 99 % accuracy, Random Forest(RF) give 97 % accuracy, Decision Tree gives 98 % accuracy, K-Nearest Neighbor(K-NN) gives 94% accuracy. The experimental procedure concludes that advances in machine learning and predictive analytics, represent a promising model to recognize intelligent solutions, which in turn prove the ability of predication in kidney disease.

TEAM MEMBER DETAILS

Vtu19278 K Mohithsai
Vtu19696 T Vishnu Vardhan Reddy
Vtu19277 K Saketh
9966753506
9866347267
9912820162
vtu19278@veltech.edu.in
vtu19696@veltech.edu.in
vtu19277@veltech.edu.in

INTRODUCTION

Chronic kidney disease (CKD) is a significant public health problem for worldwide, especially for a low and medium-income countries. CKD means that the kidney does not work and cannot correctly filter the blood. About 10 percent of the population for a worldwide suffering from (CKD), and millions of die each year because of they couldn't get the affordable treatment, with the number increasing in the elderly.

Chronic kidney disease (CKD) is a progressive condition that affects the kidneys, which are responsible for filtering waste products and excess fluids from the blood. CKD can be caused by a range of factors including high blood pressure, diabetes, autoimmune diseases, and inherited genetic conditions. As CKD progresses, the kidneys become less able to perform their filtering function, leading to a buildup of waste products and fluids in the body. This can cause a range of symptoms including fatigue, weakness, nausea, and swelling in the legs and feet.

METHODOLOGIES

the kideny disease prediction. Intially the data set is collected from the University of California, Irvine (UCI). Then that data set is preprocessed to remove the null values and repeated values. Then the output is cleaned dataset then that particular dataset is divided into two different types like training dataset and testing dataset it under goes to the machine learning algorithms like K-Nearest Neighbor,Random Forest,Decision Tree and Light Gradient Boosted Machine. By using these four algorithms the disesase is predicted and performance evaluation is done.

The training dataset and testing dataset are used in the machine learning algorithms for predicting the CKD.Based on the performance evaluation, the kidney disease can be predicted. It has experimented on UCI dataset which contains early stages of 400 CKD patients with 25 attributes.

RESULTS

As we have mentioned the earlier that it is better to search for an alternative approach with promising results and we are going to do that. So, in this study, we are going to compare four very commonly known machine learning algorithms and we are going to state the best among them using a case study on Chronic Kidney disease Prediction. As stated, the four machine learning algorithms are: Random Forest Decision Tree LGBM K- Near Neighbour (KNN) Why we are going with only four algorithms is that we don't want to overload and confuse the learning procedure by including more machine learning algorithms. So, we thought that four would be an optimal choice and the machine would work just fine. Using confusion matrix we received the accuracy of all four methodologies such as K-Nearest Neighbor, Random Forest, Decision Tree and Light Gradient Boosted Machine.

Table 1. Label in 20pt Calibri.

Algorithms names	Accuracy	Precision	Re call
K neighbours classifier	0.94	1.0	0.97
Decision Tree	0.98	1.0	1.0
LGBM Classifier	1.0	1.0	1.0
Random Forest Classifier	0.97	0.97	1.0

STANDARDS AND POLICIES

As a complementary solution, we could build a mobile application, to make it available to everyone to detect whether they are CKD or not and as of now if we have a correct data set we could use it to predict many more diseases and also if the application collaborates with a highly reputed company then we can allow making more advancements to the existing model.

```
Training Accuracy of Decision Tree Classifier is 1.0
Test Accuracy of Decision Tree Classifier is 1.0

Confusion Matrix :-
[[33  0]
 [ 0 25]]

Classification Report :-
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        33
           1       1.00      1.00      1.00        25

    accuracy                           1.00        58
   macro avg       1.00      1.00      1.00        58
weighted avg       1.00      1.00      1.00        58
```

CONCLUSIONS

CKD means that the kidney does not work and cannot correctly filter the blood. About 10 percent of the population for a worldwide suffering from (CKD), and millions of die each year because of they couldn't get the affordable treatment, with the number in- creasing in the elderly. In 2010, a study was conducted by International Society of Numerology(ISN) on global burden disease.

ACKNOWLEDGEMENT

Dr. C. KOTTESWARAN, M.E,PhD
ASSOCIATE PROFESSOR,
+91 9944194641

Figure 9.1: **Poster**

References

[1] Imesh Udara Ekanayake Damayanthi Herath , ”Chronic Kidney Disease Prediction Using Machine Learning Methods,” in 2020 IEEE Moratuwa Engineering Research Conference, 2020 .

[2] Yedilkhan Amirgaliyev Shahriar Shamiluulu Azamat Serek, ”Analysis of Chronic Kidney Disease Dataset by Applying Machine Learning Methods”,International Journal of Computing and Business Research (IJCBR), vol. 6, no. 2, (2019).

[3] T Shaikhina, Torgyn, et al. ”Implementation ofMachine LearningModels for the Prevention of Kidney Diseases (CKD) or Their Derivatives” Hindawi Computational Intelligence and Neuroscience Volume 2021, Article ID 3941978, 8 pages https://doi.org/10.1155/2021/3941978 (2021).

[4] PANKAJ CHITTORA ZBIGNIEW LEONOWICZ SANDEEP CHAURASIA, “Prediction of Chronic Kidney Disease - A Machine Learning Perspective”,Article in IEEE Access. D

[5] Jing Xiao1 Ruifeng Ding Xiulin Xu Haochen Guan , et al. ”Comparison and development of machine learning tools in the prediction of chronic kidney disease progression.” Xiao et al. J Transl Med 17:119 https://doi.org/10.1186/s12967-019-1860-0 (2019).

[6] Yesubabu Kakitapallia Janakiram Ampolua Satya Dinesh Madasub, et al., ”Detailed Review of Chronic Kidney Disease” International Publishing, Kidney Dis DOI: 10.1159/000504622. (2019).

[7] Reshma S Salma Shaji S R Ajina Vishnu Priya S R,”Chronic Kidney Disease Prediction using Machine Learning”,International Journal of Engineering Research Technology (IJERT) ISSN: 2278-0181 IJERTV9IS070092 Vol. 9 Issue 07, July-2020.

[8] S.Revathy B.Bharathi et al, ”Chronic Kidney Disease Prediction using Machine Learning Models ,” International Journal of Engineering and Advanced Technology (IJEAT) ISSN: 2249 – 8958, Volume-9 Issue-1, Oct.

[9] B.Sasi Varna G.Pravallika, ”CHRONIC KIDNEY DISEASE METHODOLOGY BY USING MACHINE LEA RNING,” IJRTI Volume 7, Issue 6 ISSN: 2456-3315(2021).

[10] Dibaba Adeba Debal and Tilahun Melak Sitote, “Chronic kidney disease prediction using machine learning techniques,” Informatics in Medicine Unlocked,Debal and Sitote Journal of Big Data 9:109 https://doi.org/10.1186/s40537-022-00657-5 (2022).