| | |
|---|---|
| **Date** | **18 November 2022** |
| **Team ID** | **PNT2022TMID48762** |
| **Project N ame** | **Early Detection Of Chronic Kidney Disease Using Machine Learning** |
| **Team Members** | **Gayathri.R , Saritha.T, Kavipriya.M** |

# CHAPTER 1

## INTRODUCTION :

Chronic Kidney Disease (CKD) is considered as an important threat for the society with respect to the health in the present era. Chronic kidney disease can be detected with regular laboratory tests, and some treatments are present which can prevent development, slow disease progression, reduce complications of decreased Glomerular Filtration Rate(GFR) and risk of cardiovascular disease, and improve survival and quality of life. CKD can be caused due to lack of water consumption, smoking, improper diet, loss of sleep and many other factors. This disease affected 753 million people globally in 2016 in which 417 million are females and 336 million are males. Majority of the time the disease is detected in its final stage and which sometimes leads to kidney failure Chronic kidney disease (CKD) is a significant public health problem worldwide, especially for low and mediumincome countries. Chronic kidney disease (CKD) means that the kidney does not work as expected and cannot correctly filter blood. About 10% of the

population worldwide suffers from (CKD), and millions die each year because they cannot get affordable treatment, with the number increasing in the elderly. According to the Global Burden Disease 2010 study conducted by the International Society of Nephrology, chronic kidney disease (CKD) has been raised as an important cause of mortality worldwide with the number of deaths increasing by 82.3% in the last two decade kidney disease (CKD) is regarded as a serious hazard. Regular laboratory testing can identify chronic kidney disease, and there are treatments available to stop the illness from progressing, lessen the problems of reduced Glomerular Filtration Rate (GFR), lowers the risk of cardiovascular disease, and enhance quality of life and survival. Lack of water intake, smoking, a poor diet, lack of sleep, and numerous other factors can lead to CKD. Globally, this illness impacted 753 million people in 2016, 417 million of them were female and 336 million male

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.

The human body has two kidneys located at the back of the peritoneal cavity, which are vital organs necessary for its proper functioning. The main function of the kidneys is to regulate the balance of salt, water and other ions and trace elements in the human body, such as calcium, phosphorus, magnesium, potassium, chlorine and acids. At the same time, the kidneys secrete hormones such as erythropoietin, vitamin D and renin. More specifically, erythropoietin stimulates the production and maturation of red blood cells in the bone marrow, while vitamin D regulates calcium and phosphorus in the body, bone structure and many other actions. The kidneys are also the site of the action of hormones that are responsible for regulating blood pressure, fluid balance or bone metabolism and vascular calcifications. Finally, the kidneys eliminate all the useless products of etabolism, as well as drugs and other toxins that enter the body.

Diabetes and high blood pressure are the two main causes of chronic kidney disease. Diabetes is characterized by high blood sugar levels, causing damage to the kidneys and heart, blood vessels and eyes. Moreover, poor control of high blood pressure can be a major cause of heart attack, stroke and chronic kidney disease. Other conditions that affect thekidneys are glomerulonephritis, hereditary diseases, dysplasia,

kidney stones, tumours, recurrent urinary tract infections, metabolic diseases, obesity and age Early diagnosis and treatment of CKD is a serious challenge for the medical community.

The treating physician (nephrologist) is called on the one hand to slow down the progression of the disease to more advanced stages, and if possible, to suspend it, and on the other hand, to treat the above-mentioned systemic manifestationsIn the current research work, a Machine Learning-based approach will be presented for CKD disease. The main contributions of the adopted methodology are the following:

• A data preprocessing step that exploits the Synthetic Minority Oversampling Technique (SMOTE), which is essential to ensure that the dataset instances are distributed in a balanced way and, thus, designs effective classification models to predict the risk for CKD occurrence.

• A features analysis, which includes three specific sub-steps: (i) numerical attributes

statistical description, (ii) order of importance measurement by employing three different methods, and (iii) capturing nominal features frequency of occurrence in tabular form.

• A comparative evaluation of various models' performance is presented considering the most common metrics, such as Precision, Recall, F-Measure, Accuracy and AUC.

• A performance evaluation is demonstrated, where all models demonstrated exceptionally high outcomes, with Rotation Forest

achieving the highest results in all metrics, thus constituting the main suggestion of this analysis.

Two bean-shaped organs, named kidney, are two important parts in human body. Kidney removes waste from blood by filtering. If this filtering system is hampered, protein can seep to urine and waste elements can remain in blood. And gradually, kidney loses its ability to filter. This failure of kidney is called Chronic Kidney Disease (CKD), also known as Chronic Renal Disease. Whole body is affected by kidney failure. Generally people suffer with this disease with their age, but recently from 5 years children and youth are also suffering from CKD disease. There are some symptoms which shows kidneys are beginning to fail like muscle cramps, nausea and vomiting, appetite losses, swelling in your feet and ankles, too much urine or not enough urine, trouble catching your breath, trouble sleeping, fever and vomiting. Risk factors of CKD are diabetes, smoking, lack of sleeping, hypertension, improper diet, etc. Among them diabetes is the more dangerous factor. At the last stage, the patient must take dialysis or do kidney transplantation. One of the best ways to reduce this death rate is early treatment. Therefore, early prediction and proper treatments can possibly stop, or slow the progression of this chronic disease

Chronic kidney disease is one of the Kidney disease in medical field. Kidneys are a pair of organs located toward lower back of the

body .It can be placed on either side of the spine. Main function of the Kidney act as a filtration system for blood and to remove toxins from body. The kidney shifts the toxins to the bladder then it later removed from the body through urination. Kidney failure occurs when the kidneys unable to filter waste from the blood.If kidneys cannot perform their regular job then body becomes overloaded with toxins. This can lead to kidney failure and can result in death. Kidney failure suffers from one or more of the following causes: Loss of Blood Flow to the Kidneys, Damage to the Kidneys and Urine Elimination Problems. Kidney problems can be either acute or chronic (fig:1). Acute kidney disease is the sudden loss of kidney function that occurs when high levels of waste products of the body's metabolism accumulate in the blood.

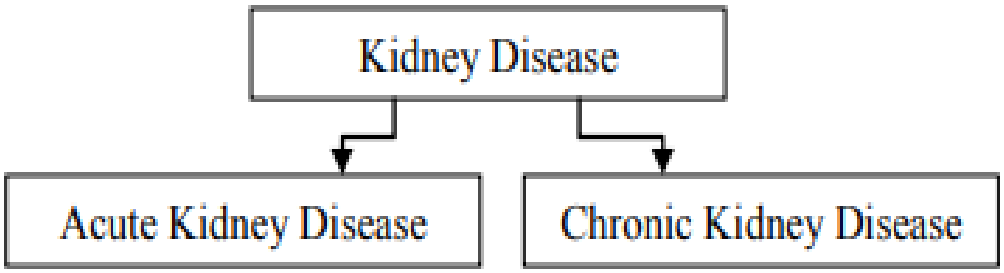
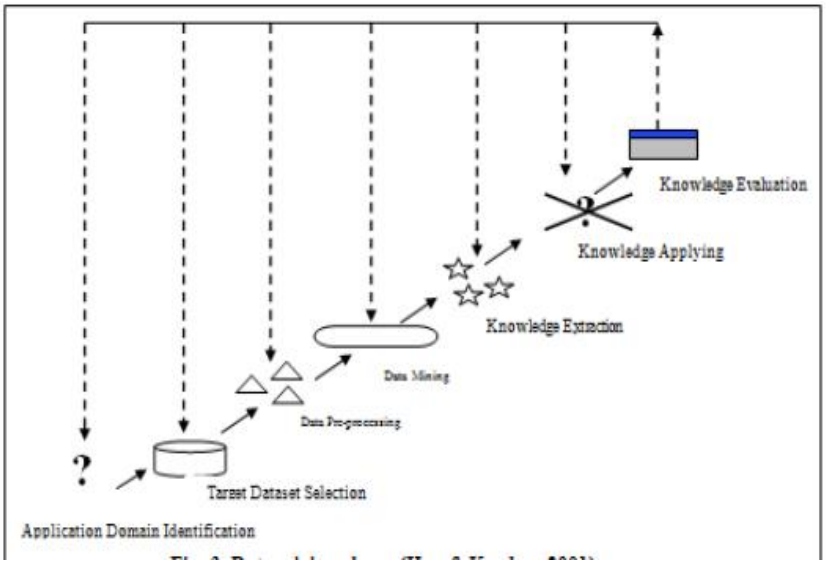



**Fig:1 Types of kidney disease**

Chronic kidney disease is a gradual development of permanent kidney disease.It is the most common type of kidney disease and occurs when the kidneys are damaged or are not functioning for some months or longer. Some of the leading causes of chronic kidney disease are diabetes, hypertension, lupus and complications from some medications. Medications can control hypertension and diabetes and changes in diet and lifestyle. Complications of chronic kidney disease such as anemia and weak bones leading to fractures.Chronic kidney disease includes a number of conditions affecting function of kidney.Many people may be in the early stages of kidney disease and not have any indication .There are certain symptoms,as follows for chronic kidney disease.

Diabetes

- High blood pressure
- Coronary artery Disease
- Anemia
- Bacteria and albumin in urine
- Deficiency of Sodium and Potassium in blood and Family history of kidney disease.


Knowledge Evaluation
Knowledge Applying
Knowledge Extraction
Data Mining
Data Pre-processing
Target Dataset Selection
Application Domain Identification

# CHAPTER 2

# SYSTEM ANALYSIS

## EXISTING SYSTEM:

To predict diseases, data mining or machine learning models are playing a vital role. By aking some mathematical approaches, data mining models extract patterns from data and later these patterns are used for the survival of patients. Multilayer Perceptron (MP), Support Vector Machine (SVM), KNearest Neighbour (KNN), Logistic Regression (LR), Naïve Bayes (NB), Random Forest (RF), etc. are some renowned machine learning methods which were successfully implemented to examine and classify the kidney disease. IN recent times, some researchers have been working on CKD by applying different computational techniques for the prediction and diagnosis of this disease.

## DISADVANTAGES :

Machine learning algorithms can build complex models and make accurate decisions when given relevant data. When there is an adequate amount of data, the performance of machine learning algorithms is expected to be sufficiently satisfactory. However, in specific applications, the data are often insufficient. Therefore, it is

important to analyse these algorithms and obtain good results with a relatively small sample size.

## 2.2 PROPOSED SYSTEM:

In proposed sydtem we using dataset we are using mean, mode and median based pre-processing techniques for the missing values. Further, we have used K-Nearest Neighbour Classifier, Decision Tree Classifier, Gaussian Naïve Bayes, Logical Regression and Artificial Neural Network to train the model. Then, based on the results of each of these Machine Learning Methods, we can compare and determine which among the following methods can predict the possibility of CKD most accurately

# CHAPTER 3

## 4.1 SYSTEM SPECIFICATION

### HARDWARE REQUIREMENTS:

- System : Pentium IV 2.4 GHz.
- Hard Disk : 40 GB.

- Monitor : 14’ Colour Monitor.
- Mouse : Optical Mouse.
- Ram : 512 Mb.

## 4.2 SOFTWARE REQUIREMENTS:

- Operating system : Windows 10
- Coding Language : python with C#

# CHAPTER 5

# SOFTWARE DESCRIPTION

Anaconda is an open-source software that contains Jupyter, spyder, etc that are used for large data processing, data analytics, heavy scientific computing. Anaconda works for R and python programming language. Spyder(sub-application of Anaconda) is used for python. Opencv for python will work in spyder. Package versions are managed by the package management system called conda.Anaconda is a distribution of the Python and R programming languages for scientific computing (data science, machine learning applications, large-scale data processing, predictive analytics, etc.), that aims to simplify package management and deployment. The distribution includes data-science packages suitable for Windows, Linux, and macOS. It is developed and maintained by Anaconda, Inc., which was founded by Peter Wang and Travis Oliphant in 2012.[8] As an Anaconda, Inc. product, it is also known as Anaconda Distribution or Anaconda Individual Edition, while other products from the company are Anaconda Team Edition and Anaconda Enterprise Edition, both of which are not free.[

Python is a widely used high-level, general-purpose, interpreted, dynamic programming language. Its design philosophy emphasizes code readability, and its syntax allows programmers to express concepts in fewer lines of code than would be possible in languages such as C++ or Java. The language provides constructs intended to enable clear programs on both a small and large scale.

Anaconda Navigator is a desktop graphical user interface (GUI) included in Anaconda distribution that allows users to launch applications and manage conda packages, environments and channels without using command-line commands. Navigator can search for packages on Anaconda Cloud or in a local Anaconda Repository, install them in an environment, run the packages and update them. It is available for Windows, macOS and Linux.

Why learn Python? Python is an object-oriented, interpreted, mid-level programming language that is easy to learn and use while being versatile enough to tackle a variety of tasks (Helmus & Collis, 2016). It's open-source nature has skyrocketed its popularity since its first appearance in 1991 (Van Rossum & Drake Jr, 1995), and is now considered among the top programming languages to learn (Saabith, Fareez, & Vinothraj, 2019). It is free to use, has cross-platform compatibility (Mac, Windows, Linux, Ubuntu) and has low system requirements, giving any individual the opportunity to code in Python. It already has an immense community ranging from everyday individuals to top research scientists who have developed interesting projects related to data science, machine learning, artificial intelligence, app and game development, scientific research and more. These projects are easy to find thanks to the open-source community that continues to develop the language capabilities and can be searched by simply adding the key word "Python" to any inquiry.

This community also provides a wealth of resources such as courses, source codes, solutions to com

What is Anaconda? Anaconda is a free software that provides you with a toolkit that is tailored for research and science. Installing Anaconda gives you access to different environments that allow you to code in either Python or R. These environments, also known as integrated development environments (IDEs), are platforms or apps that greatly ease the development of code. They serve a similar role to text processors like Microsoft Word, Google Docand Pages for writing text, but in truth they are so much more. IDEs contain many useful features to write, edit and debug code, visualize and inspect data, store variables, present results, and collaborate on projects. While the display and the quirks of an IDE differ, the programming language does not. Thus, changing IDEs in Anaconda does not result in drastically changing your Python code. The learning curve is mostly related to understanding Python's syntax. Once you learn how to code in one IDE, you will be able to transfer this skill to another with ease. One IDE is not necessarily better than another, each comes with its own pros and cons and which one you choose comes down to personal preference. Downloading the toolkit also provides you with an enormous selection of prebuilt functions that the Python community has coded in the past. These functions are regrouped in what are called libraries and can be downloaded easily through Anaconda. There are many different ways to

install and use Python on your computer, but Anaconda is a simple, well-supported, graphical user interface (GUI) that includes the most important libraries and IDEs in its installation. Anaconda also simplifies the process of keeping all of these libraries up to date. Thus, rather than installing Python separately with different IDEs, libraries and functionalities, Anaconda can do it for you in one installation

**CONDA:**

Conda is an open-source package and environment management system that runs on Windows, macOS, and Linux. Conda quickly installs, runs, and updates packages and their dependencies. It also easily creates, saves, loads, and switches between environments on your local computer. It was created for Python programs, but it can package and distribute software for any language

**Download and install Anaconda:**

Head over to anaconda.com and install the latest version of Anaconda. Make sure to download the “Python 3.7 Version” for the appropriate architecture**.**

## PACKAGES AVAILABLE IN ANACONDA

- Over 250 packages are automatically installed with Anaconda.

- Over 7,500 additional open-source packages (including R) can be individually installed from the Anaconda repository with the `conda install` command.
- Thousands of other packages are available from Anaconda.org.
- You can download other packages using the `pip install` command that is installed with Anaconda. Pip packages provide many of the features of conda packages and in some cases they can work together. However, the preference should be to install the conda package if it is available.
- You can also make your own custom packages using the `conda build` command, and you can share them with others by uploading them to Anaconda.org, PyPI, or other repositories.

**Why use venv-based virtual environments**

- You prefer their workflow or spec formats.
- You prefer to use the system Python and libraries.
- Your project maintainers only publish to PyPI, and you prefer packages that come more directly from the project maintainers, rather than someone else providing builds based on the same code.

**Why use conda virtual environments?:**

- You want control over binary compatibility choices.

- You want to utilize newer language standards, such as C++ 17.
- You need libraries beyond what the system Python offers.
- You want to manage packages from languages other than Python in the same space.

**Workflow differentiators:**

Some questions to consider as you determine your preferred workflow and virtual environment:

- Is your environment shared across multiple code projects?
- Does your environment live alongside your code or in a separate place?
- Do your install steps involve installing any external libraries?
- Do you want to ship your environment as an archive of some sort containing the actual files of the environment?

**Package system differentiators:**

There are potential benefits for choosing PyPI or conda.

PyPI has one global namespace and distributed ownership of that namespace. Because of this, it is easier within PyPI to have single sources for a package directly from package maintainers.

Conda has unlimited namespaces (channels) and distributed ownership of a given channel. As such, it is easier to ensure binary compatibility within a channel using conda.

Downloading and Installing Anaconda The Anaconda website provides the necessary links to download the software for free in either Windows, Mac, or Linux. It also has starter videos, documentation, training, and support. To simplify this process, this tutorial has provided a step-by-step installation guide as well as a summary of the different options you may encounter. To download and install Anaconda, you must first select your operating system, desired version of Python, and whether you want 32-Bit or 64-Bit Graphical Installer which depends on your system's processor type (see Fig.1) and click on the appropriate link from their website to begin the download: Official Anaconda Website for Downloads. It is important to notice that Python 3 is the latest version and is being constantly updated and supported. The previous version, Python 2 is still offered, but is no longer being maintained. Consequently, this tutorial will focus on Python 3. Once the download has finished, open the Anaconda setup. If you are using a Windows operating system please follow Fig. 2 and if you are using a Mac operating system follow Fig. 3. For the Windows installation in the "Advanced installation options" (Fig. 2, panel e) , we recommend not selecting either option for first time users. We encourage anyone

interested in using these options to do their research to ensure they understand the pros and cons

Getting Started with Programming As previously mentioned, there are many environments that can be used to program in Python. That being said, all of them share similar components, despite being displayed differently. These common features are the Console and Editor. The difference between these two are the following: the editor permits you to write and save any code/- variables as a script and acts as a long-term memory. The console is like a short-term/working memory that allows the output to be displayed but nothing is saved. Thus, the console is used to test certain lines of code to ensure that they work, such as a program to calculate reaction time. The editor is used to save your work in memory, such as a script containing the entire code for your experimental paradigm. In figure 5, the console and editors are represented in all three IDEs (Spyder, Jupyter Notebook, and JupyterLab). Note that the layout differs, but in each IDE, the core features remain the same. It is important to also mention that for each IDE you encounter, certain features are added, making them best suited for specific tasks. Depending on your intentions, it is worth evaluating which

Open Source Innovation • Access open-source packages—such as conda, CRAN, and standard Python. • Add custom proprietary

packages to your own mirrored enterprise repository. • Open-source packages are carefully vetted and curated in our professional repository by the experts at Anaconda.

Secure Pipeline • Create differentiated channels that adhere to your organization's standards by filtering, managing, and securing select packages for every project. • Control which packages your team can download and who can access them. • Keep vulnerabilities and unreliable software out of your data science and machine learning pipeline.

**Simple Workflows :**

• Distribute packages across user channels and give users quicker access to the open-source software they need through your dedicated server. • Packages are delivered automatically to your workflow. Be in control of the quality of artifacts your team uses. When you mirror our repository for data science and machine learning packages onto your own infrastructure, you're in control of the quality of artifacts your team uses in enterprise projects. The experts at Anaconda are carefully vetting and curating CVEs so that your organization can fully leverage the power of open-source software.

# CHAPTER 6

# SYSTEM DESIGN

## 6.1 INPUT DESIGN:

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:'

- What data should be given as input?
- How the data should be arranged or coded?
- The dialog to guide the operating personnel in providing input.
- Methods for preparing input validations and steps to follow when error occur.

## OBJECTIVES

1.Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.

2. It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.

3.When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user

will not be in maize of instant. Thus the objective of input design is to create an input layout that is easy to follow

## 6.2 OUTPUT DESIGN:

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system's relationship to help user decision-making.

1. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.

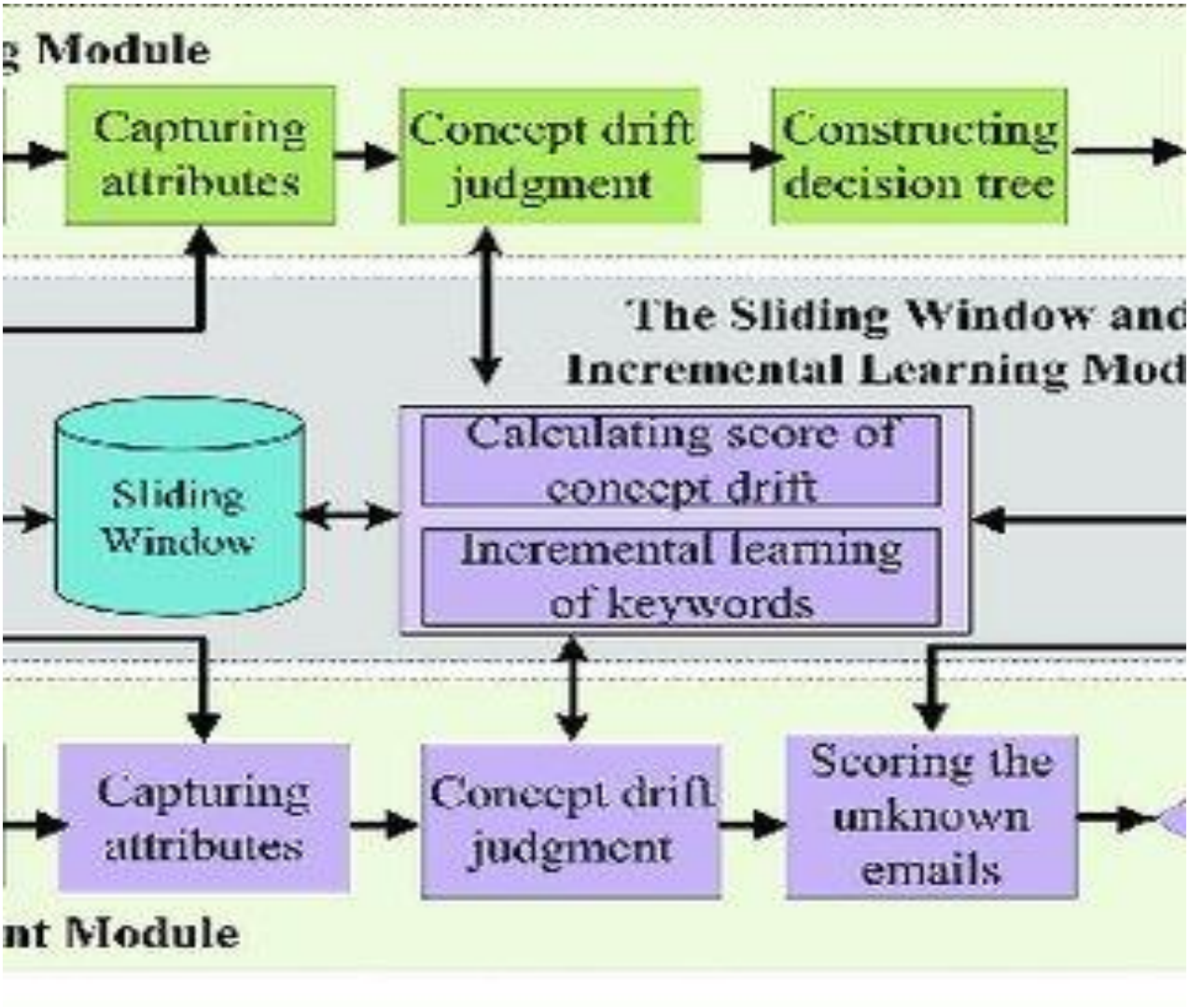
2.Select methods for presenting information.

3.Create document, report, or other formats that contain information produced by the system.

The output form of an information system should accomplish one or more of the following objectives.

- Convey information about past activities, current status or projections of the
- Future.
- Signal important events, opportunities, problems, or warnings.
- Trigger an action.
- Confirm an action.

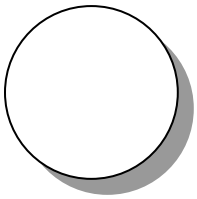
## 6.3 SYSTEM ARCHITECTURE:



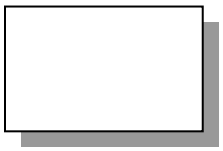
## 6.4 DATA FLOW DIAGRAM:

Data flow oriented techniques advocate that the major data items handled by a system must be first identified and then the processing required on these data items to produce the desired outputs should be determined. The DFD (also called as bubble chart) is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on these data, and the output generated by the system. It was introduced by De Macro (1978), Gane and Sarson (1979).The primitive symbols used for constructing DFD's are:

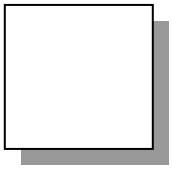
**Symbols used in DFD**



A *circle* represents a process.



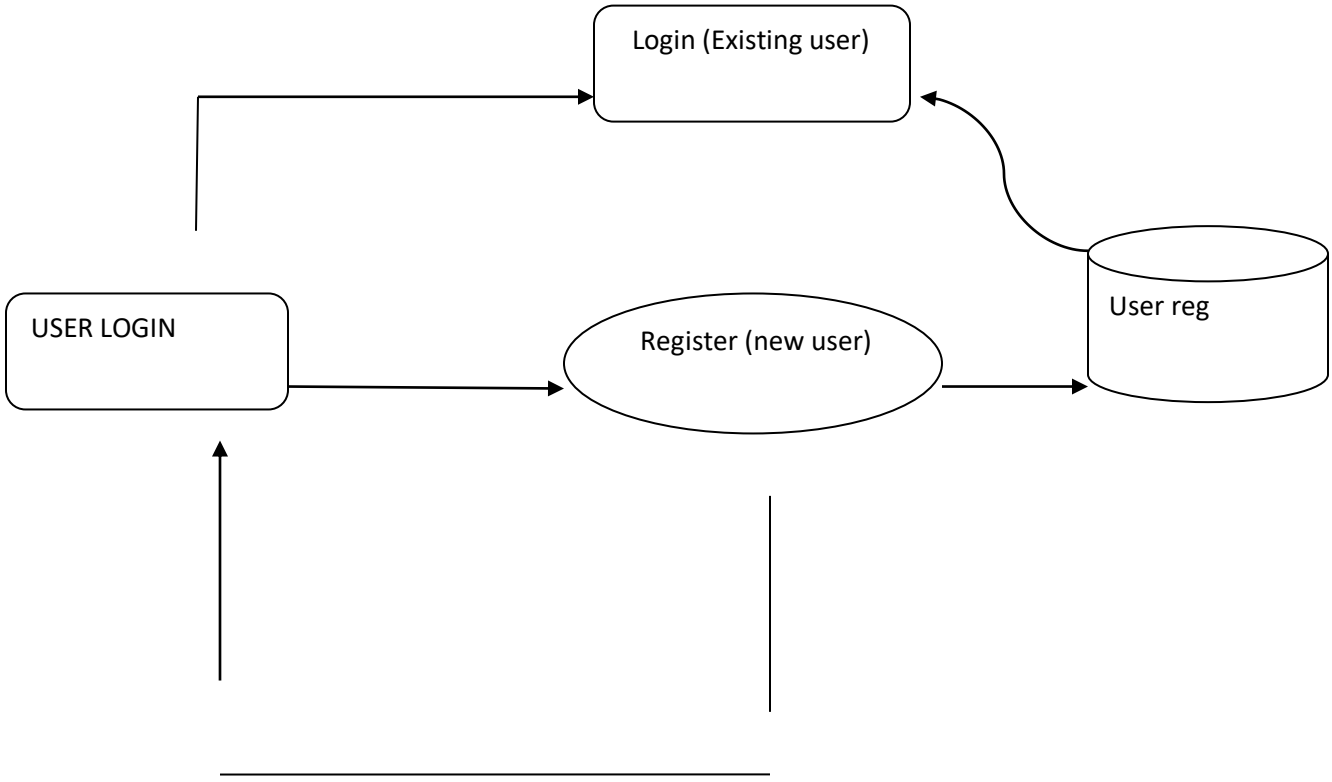
A *rectangle* represents external entity



A *square* defines a source or destination of the system data.

*Double line* with one end closed indicates data store

**LEVEL 0:**



# CHAPTER 5

## SYSTEM TESTING

System testing is the stage of implementation, which is aimed at ensuring that the system works accurately and efficiently before live operation commences. Testing is the process of executing the program with the intent of finding errors and missing operations and also a complete verification to determine whether the objectives are met and the user requirements are satisfied. The ultimate aim is quality assurance.Tests are carried out and the results are compared with the expected document. In the case of erroneous results, debugging is done. Using detailed testing strategies a test plan is carried out on each module.

### Unit Testing

The software units in a system are modules and routines that are assembled and integrated to perform a specific function. Unit testing focuses first on modules, independently of one another, to locate errors. This enables, to detect errors in coding and logic that are contained within each module.

This testing includes entering data and ascertaining if the value matches to the type and size supported by java. The various controls are tested to ensure that each performs its action as required.

Integration Testing

Data can be lost across any interface, one module can have an adverse effect on another, sub functions when combined, may not produce the desired major functions. Integration testing is a systematic testing to discover errors associated within the interface. The objective is to take unit tested modules and build a program structure. All the modules are combined and tested as a whole. Here the Server module and Client module options are integrated and tested.This testing provides the assurance that the application is well integrated functional unit with smooth transition of data.

User Acceptance Testing**:**

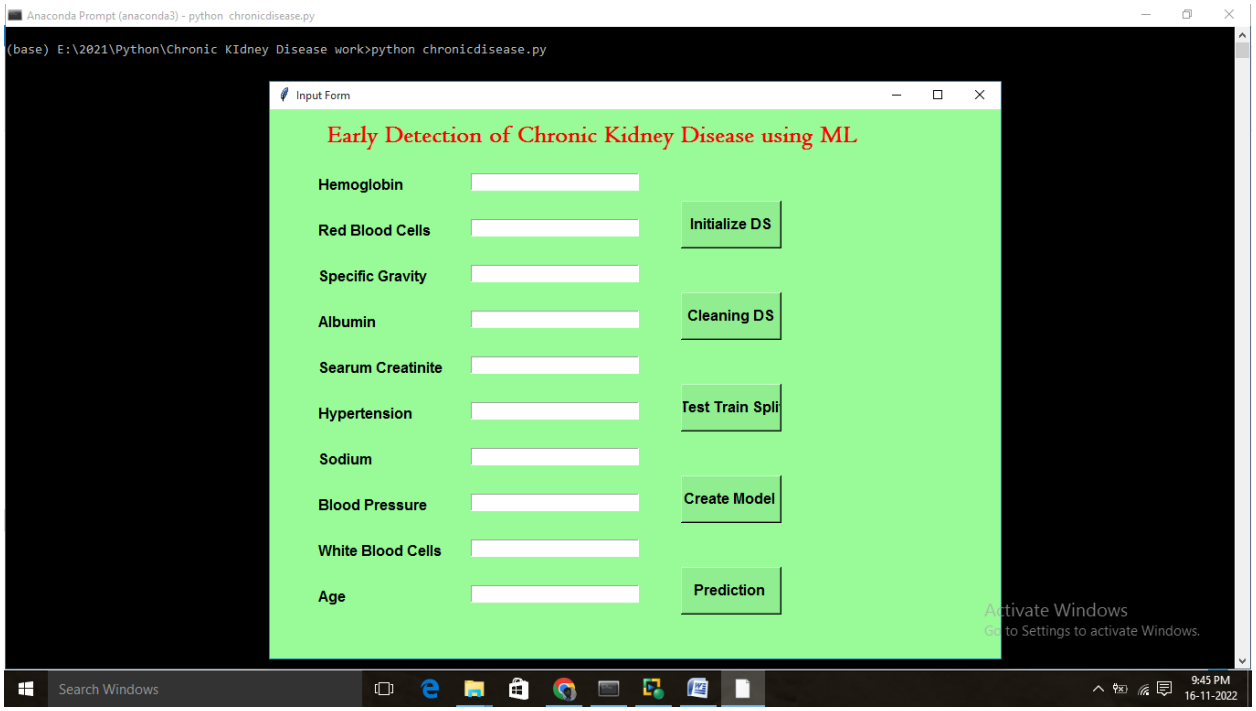
User acceptance of a system is the key factor for the success of any system. The system under consideration is tested for user acceptance by constantly keeping in touch with the system users at time of developing and making changes whenever required.

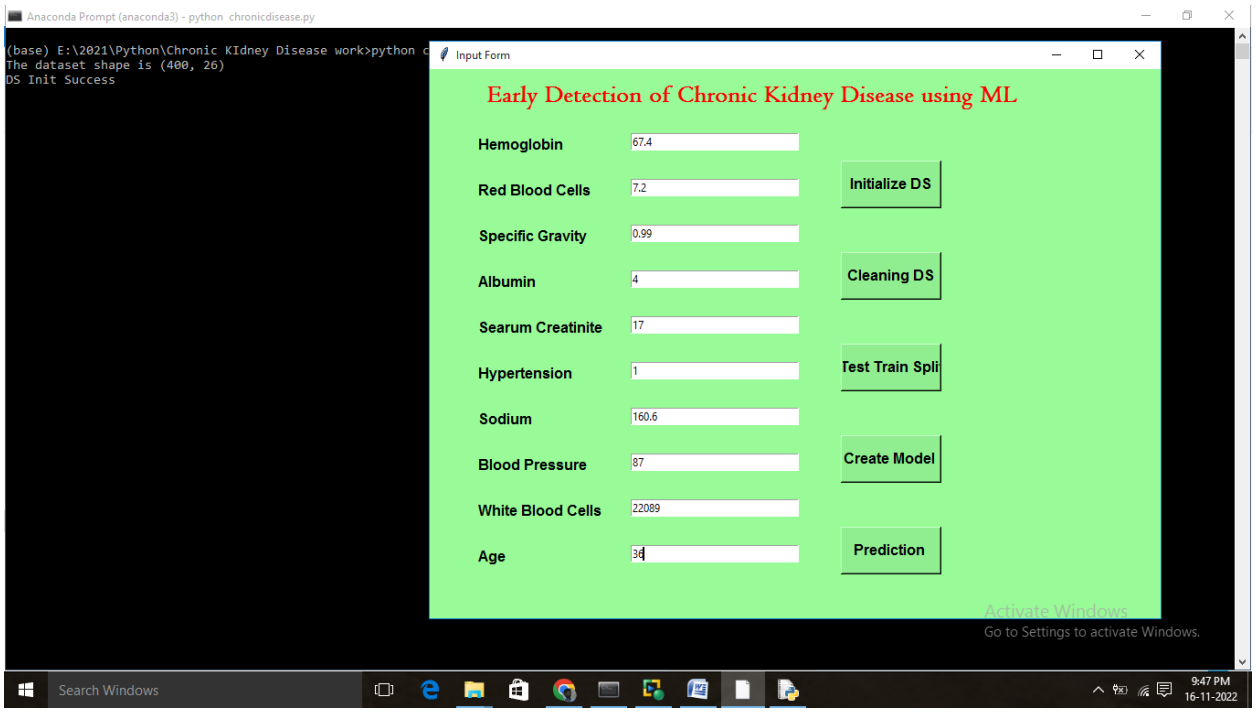
**White Box Testing:**

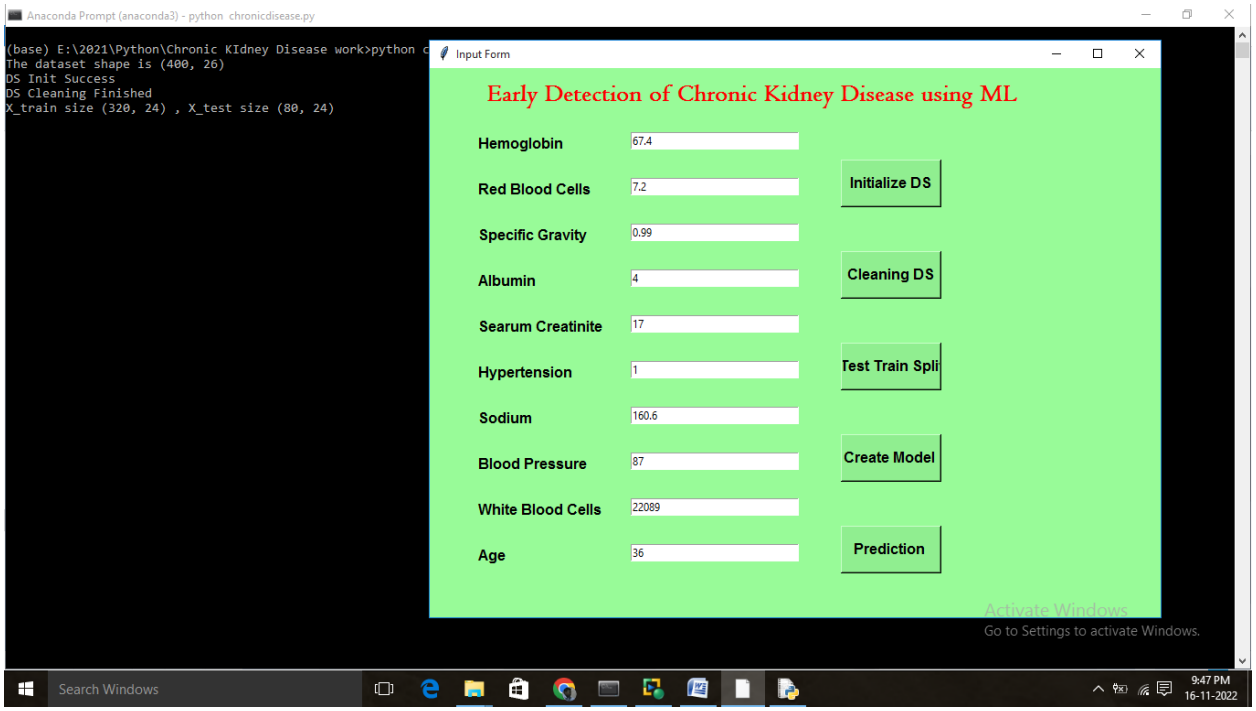
White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

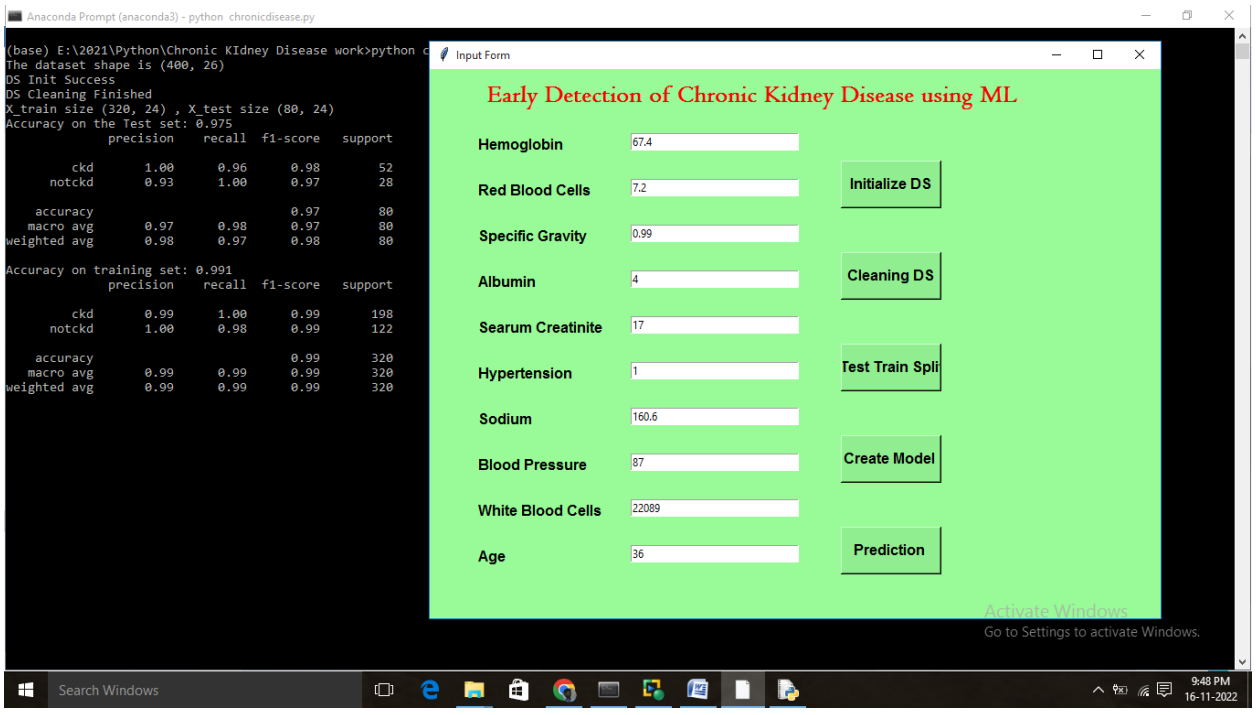
**Black Box Testing:**

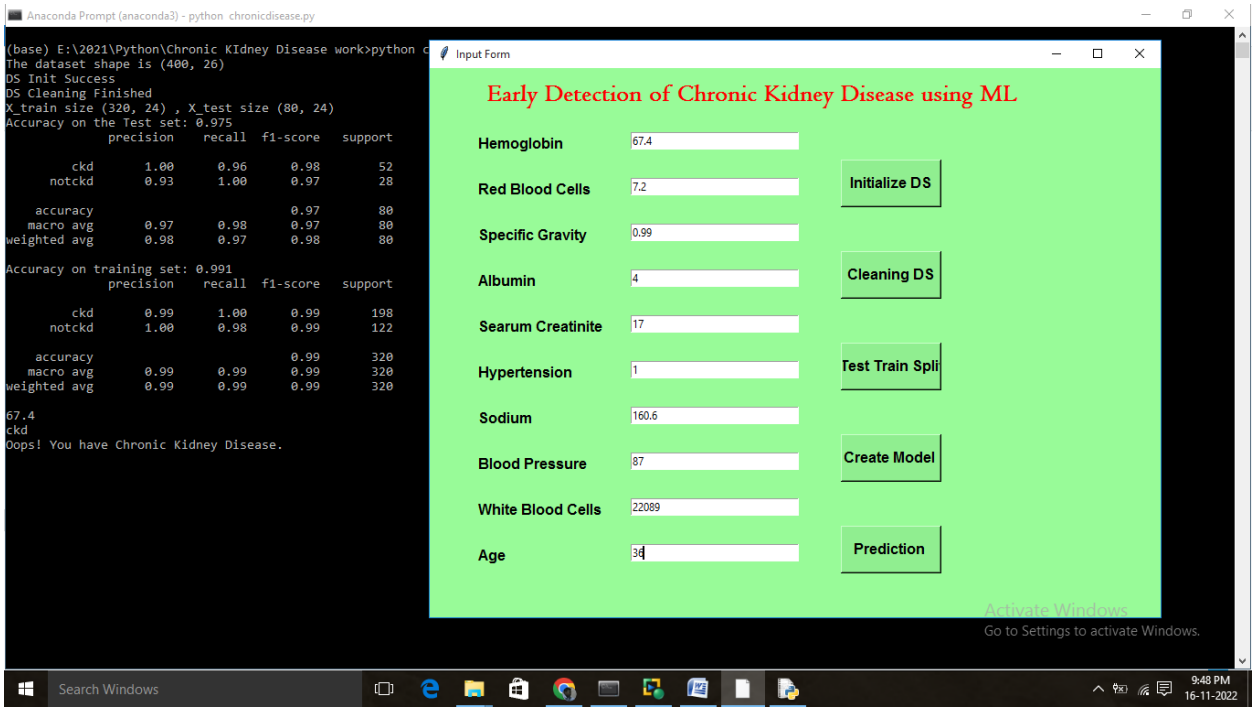
Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot "see" into it. The test provides inputs and responds to outputs without considering how the software works.


Anaconda Prompt (anaconda3) - python chronicdisease.py
(base) E:\2021\Python\Chronic KIdney Disease work>python chronicdisease.py
Input Form
Early Detection of Chronic Kidney Disease using ML
Hemoglobin
Red Blood Cells
Specific Gravity
Albumin
Searum Creatinite
Hypertension
Sodium
Blood Pressure
White Blood Cells
Age
Initialize DS
Cleaning DS
Test Train Spli
Create Model
Prediction
Activate Windows
Go to Settings to activate Windows.
Search Windows
9:45 PM
16-11-2022


Anaconda Prompt (anaconda3) - python chronicdisease.py
(base) E:\2021\Python\Chronic KIdney Disease work>python c
The dataset shape is (400, 26)
DS Init Success
Input Form
Early Detection of Chronic Kidney Disease using ML
Hemoglobin 67.4
Red Blood Cells 7.2
Specific Gravity 0.99
Albumin 4
Searum Creatinite 17
Hypertension 1
Sodium 160.6
Blood Pressure 87
White Blood Cells 22089
Age 36
Initialize DS
Cleaning DS
Test Train Spli
Create Model
Prediction
Activate Windows
Go to Settings to activate Windows.
Search Windows
9:47 PM
16-11-2022


Anaconda Prompt (anaconda3) - python chronicdisease.py
(base) E:\2021\Python\Chronic KIdney Disease work>python c
The dataset shape is (400, 26)
DS Init Success
DS Cleaning Finished
X_train size (320, 24) , X_test size (80, 24)
Input Form
Early Detection of Chronic Kidney Disease using ML
Hemoglobin
67.4
Red Blood Cells
7.2
Specific Gravity
0.99
Albumin
4
Searum Creatinite
17
Hypertension
1
Sodium
160.6
Blood Pressure
87
White Blood Cells
22089
Age
36
Initialize DS
Cleaning DS
Test Train Spli
Create Model
Prediction
Activate Windows
Go to Settings to activate Windows.
Search Windows
9:47 PM
16-11-2022


Anaconda Prompt (anaconda3) - python chronicdisease.py
(base) E:\2021\Python\Chronic KIdney Disease work>python c
The dataset shape is (400, 26)
DS Init Success
DS Cleaning Finished
X_train size (320, 24) , X_test size (80, 24)
Accuracy on the Test set: 0.975
precision recall f1-score support
ckd 1.00 0.96 0.98 52
notckd 0.93 1.00 0.97 28
accuracy 0.97 80
macro avg 0.97 0.98 0.97 80
weighted avg 0.98 0.97 0.98 80
Accuracy on training set: 0.991
precision recall f1-score support
ckd 0.99 1.00 0.99 198
notckd 1.00 0.98 0.99 122
accuracy 0.99 320
macro avg 0.99 0.99 0.99 320
weighted avg 0.99 0.99 0.99 320
Input Form
Early Detection of Chronic Kidney Disease using ML
Hemoglobin
67.4
Red Blood Cells
7.2
Specific Gravity
0.99
Albumin
4
Searum Creatinite
17
Hypertension
1
Sodium
160.6
Blood Pressure
87
White Blood Cells
22089
Age
36
Initialize DS
Cleaning DS
Test Train Spli
Create Model
Prediction
Activate Windows
Go to Settings to activate Windows.
Search Windows
9:48 PM
16-11-2022


Anaconda Prompt (anaconda3) - python chronicdisease.py
(base) E:\2021\Python\Chronic KIdney Disease work>python c
The dataset shape is (400, 26)
DS Init Success
DS Cleaning Finished
X_train size (320, 24) , X_test size (80, 24)
Accuracy on the Test set: 0.975
precision recall f1-score support
ckd 1.00 0.96 0.98 52
notckd 0.93 1.00 0.97 28
accuracy 0.97 80
macro avg 0.97 0.98 0.97 80
weighted avg 0.98 0.97 0.98 80
Accuracy on training set: 0.991
precision recall f1-score support
ckd 0.99 1.00 0.99 198
notckd 1.00 0.98 0.99 122
accuracy 0.99 320
macro avg 0.99 0.99 0.99 320
weighted avg 0.99 0.99 0.99 320
67.4
ckd
Oops! You have Chronic Kidney Disease.
Input Form
Early Detection of Chronic Kidney Disease using ML
Hemoglobin
67.4
Red Blood Cells
7.2
Specific Gravity
0.99
Albumin
4
Searum Creatinite
17
Hypertension
1
Sodium
160.6
Blood Pressure
87
White Blood Cells
22089
Age
36
Initialize DS
Cleaning DS
Test Train Spli
Create Model
Prediction
Activate Windows
Go to Settings to activate Windows.
Search Windows
9:48 PM
16-11-2022

**CODING:**

**SAMPLE CODING:**

```
from tkinter import *
import sqlite3
import re
from tkinter import messagebox
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
#import seaborn as sns
import warnings
warnings.filterwarnings('ignore')



class patient1:
# root = Tk()
```

```
 def __init__(self,root):
  self.root= root
  root.title("Input Form")
  root.geometry('500x500')
  #Button(root,
text='Reset',width=20,bg='brown',fg='white',command=validation).place
(x=180,y=430)
  # Centering Root Window on Screen


  w = 800 # width for the Tk root
  h = 600 # height for the Tk root


  # get screen width and height
  ws = root.winfo_screenwidth() # width of the screen
  hs = root.winfo_screenheight() # height of the screen


  # calculate x and y coordinates for the Tk root window
  x = (ws/2) - (w/2)
  y = (hs/2) - (h/2)
```

```
  root["bg"] = '#98fb98'
  # set the dimensions of the screen
  # and where it is placed
  root.geometry('%dx%d+%d+%d' % (w, h, x, y))



  self.v = IntVar()
  self.hb2=StringVar()
  self.rbc2=StringVar()
  self.sg2 =StringVar()
  self.al2=StringVar()
  self.sc2=StringVar()
  self.ht2=StringVar()
  self.sod2=StringVar()
  self.bp2 =StringVar()
  self.wbc2=StringVar()
  self.age2=StringVar()


# labels for the window
```

```
  self.heading = Label(self.root, text="Early Detection of Chronic
Kidney Disease using ML", font=('Centaur 20 bold'),
bg='#98fb98',fg='red')

  self.heading.place(x=60, y=10)



  self.hb1 = Label(self.root, text="Hemoglobin", font=('arial 12
bold'),bg='#98fb98')

  self.hb1.place(x=50, y=70)


  self.hb_ent = Entry(self.root, width=30, textvar=self.hb2)

  self.hb_ent.place(x=220, y=70)


  self.rbc1 = Label(self.root, text="Red Blood Cells", font=('arial 12
bold'),bg='#98fb98')

  self.rbc1.place(x=50, y=120)


  self.rbc_ent = Entry(self.root, width=30, textvar=self.rbc2)

  self.rbc_ent.place(x=220, y=120)


  self.sg1 = Label(self.root, text="Specific Gravity", font=('arial 12
bold'),bg='#98fb98')
```

```
  self.sg1.place(x=50, y=170)


  self.sg_ent = Entry(self.root, width=30, textvar=self.sg2)
  self.sg_ent.place(x=220, y=170)


  self.al1 = Label(self.root, text="Albumin", font=('arial 12
bold'),bg='#98fb98')
  self.al1.place(x=50, y=220)


  self.al_ent = Entry(self.root, width=30, textvar=self.al2)
  self.al_ent.place(x=220, y=220)


  self.sc1 = Label(self.root, text="Searum Creatinite", font=('arial 12
bold'),bg='#98fb98')
  self.sc1.place(x=50, y=270)


  self.sc_ent = Entry(self.root, width=30, textvar=self.sc2)
  self.sc_ent.place(x=220, y=270)


  self.ht1 = Label(self.root, text="Hypertension", font=('arial 12
bold'),bg='#98fb98')
  self.ht1.place(x=50, y=320)
```

```
  self.ht_ent = Entry(self.root, width=30, textvar=self.ht2)
  self.ht_ent.place(x=220, y=320)


  self.sod1 = Label(self.root, text="Sodium", font=('arial 12
bold'),bg='#98fb98')
  self.sod1.place(x=50, y=370)


  self.sod_ent = Entry(self.root, width=30, textvar=self.sod2)
  self.sod_ent.place(x=220, y=370)


  self.bp1 = Label(self.root, text="Blood Pressure", font=('arial 12
bold'),bg='#98fb98')
  self.bp1.place(x=50, y=420)


  self.bp_ent = Entry(self.root, width=30, textvar=self.bp2)
  self.bp_ent.place(x=220, y=420)


  self.wbc1 = Label(self.root, text="White Blood Cells", font=('arial 12
bold'),bg='#98fb98')
```

```
  self.wbc1.place(x=50, y=470)


  self.wbc_ent = Entry(self.root, width=30, textvar=self.wbc2)
  self.wbc_ent.place(x=220, y=470)



  self.age1 = Label(self.root, text="Age", font=('arial 12
bold'),bg='#98fb98')
  self.age1.place(x=50, y=520)


  self.age_ent = Entry(self.root, width=30, textvar=self.age2)
  self.age_ent.place(x=220, y=520)


# button to perform a command


  self.submit = Button(self.root, text="Initialize DS", font="aried 12
bold",width=10, height=2, bg='lightgreen',command=self.initds)
  self.submit.place(x=450, y=100)


  self.submit1 = Button(self.root, text="Cleaning DS", font="aried 12
bold",width=10, height=2, bg='lightgreen',command=self.logform)
  self.submit1.place(x=450, y=200)
```

```
  self.submit2 = Button(self.root, text="Test Train Split", font="aried 12
bold",width=10, height=2, bg='lightgreen',command=self.splitds)
  self.submit2.place(x=450, y=300)

  self.submit3 = Button(self.root, text="Create Model ", font="aried 12
bold",width=10, height=2, bg='lightgreen',command=self.classifyy)
  self.submit3.place(x=450, y=400)

  self.submit4 = Button(self.root, text="Prediction ", font="aried 12
bold",width=10, height=2, bg='lightgreen',command=self.predictt)
  self.submit4.place(x=450, y=500)

 def initds(self):
  self.df=pd.read_csv("kidney_disease.csv")
  print("The dataset shape is {}".format(self.df.shape))
  # remove "id" feature
  self.df.drop('id',axis=1,inplace=True)
  print("DS Init Success")

 def logform(self):
```

#in our dataset some features ['pcv','wc','rc','dm','cad','classification'] contains some special character.so replace them with appropriate values.

```
# cleaning 'PCV'

self.df['pcv']=self.df['pcv'].apply(lambda x:x if type(x)==type(3.5) else x.replace('\t43','43').replace('\t?','Nan'))

# cleaning "WC"

self.df['wc']=self.df['wc'].apply(lambda x:x if type(x)==type(3.5) else x.replace('\t?','Nan').replace('\t6200','6200').replace('\t8400','8400'))

# cleaning "RC"

self.df['rc']=self.df['rc'].apply(lambda x:x if type(x)==type(3.5) else x.replace('\t?','Nan'))

# cleaning "dm"

self.df['dm']=self.df['dm'].apply(lambda x:x if type(x)==type(3.5) else x.replace('\tno','no').replace('\tyes','yes').replace(' yes','yes'))

# cleaning "CAD"

self.df['cad']=self.df['cad'].apply(lambda x:x if type(x)==type(3.5) else x.replace('\tno','no'))
```

```
  # cleaning "Classification"

  self.df['classification']=self.df['classification'].apply(lambda x:x if
type(x)==type(3.5) else x.replace('ckd\t','ckd'))


  #Note: Some features are mistyped as "object".so convert them into
"float" type


  mistyped=[['pcv','rc','wc']]

  for i in mistyped:

   self.df[i]=self.df[i].astype('float')

  #  define categoricsl features

  cat_cols=list(self.df.select_dtypes('object'))

  cat_cols


  # define numeric features

  self.num_cols=list(self.df.select_dtypes(['int64','float64']))

  self.num_cols

  # Checking missing/Nan values

  self.df.isnull().sum().sort_values(ascending=False)


  # Let's impute Nan Values with median in numeric features
```

```
 for col in self.num_cols:
  self.df[col]=self.df[col].fillna(self.df[col].median())
 # let's impute categorical features with most frequent value
 self.df['rbc'].fillna('normal',inplace=True)
 self.df['pc'].fillna('normal',inplace=True)
 self.df['pcc'].fillna('notpresent',inplace=True)
 self.df['ba'].fillna('notpresent',inplace=True)
 self.df['htn'].fillna('no',inplace=True)
 self.df['dm'].fillna('no',inplace=True)
 self.df['cad'].fillna('no',inplace=True)
 self.df['appet'].fillna('good',inplace=True)
 self.df['pe'].fillna('no',inplace=True)
 self.df['ane'].fillna('no',inplace=True)
 self.df.isna().sum().sort_values(ascending=False)
 print("DS Cleaning Finished")

def splitds(self):

 self.df['rbc']=self.df['rbc'].map({'normal':0,'abnormal':1})
 self.df['pc']=self.df['pc'].map({'normal':0,'abnormal':1})
 self.df['pcc']=self.df['pcc'].map({'notpresent':0,'present':1})
```

```
  self.df['ba']=self.df['ba'].map({'notpresent':0,'present':1})
  self.df['htn']=self.df['htn'].map({'no':0,'yes':1})
  self.df['dm']=self.df['dm'].map({'no':0,'yes':1})
  self.df['cad']=self.df['cad'].map({'no':0,'yes':1})
  self.df['pe']=self.df['pe'].map({'no':0,'yes':1})
  self.df['ane']=self.df['ane'].map({'no':0,'yes':1})
  self.df['appet']=self.df['appet'].map({'good':0,'poor':1})

  # scaling with MinMaxScaler
  from sklearn.preprocessing import StandardScaler,MinMaxScaler
  mm_scaler=MinMaxScaler()
  self.df[self.num_cols]=mm_scaler.fit_transform(self.df[self.num_cols])

  from sklearn.model_selection import train_test_split
  x=self.df.drop('classification',axis=1)
  y=self.df['classification']

self.X_train,self.X_test,self.y_train,self.y_test=train_test_split(x,y,test_s
ize=0.2,random_state=0)
  print("X_train size {} , X_test size
{}".format(self.X_train.shape,self.X_test.shape))
```

```
 def classifyy(self):
  from sklearn.ensemble import RandomForestClassifier
  from sklearn.metrics import
confusion_matrix,classification_report,accuracy_score
  # Creating Random Forest model
  self.rf=RandomForestClassifier(max_depth=5,n_estimators=5)
  self.rf.fit(self.X_train,self.y_train)
  self.y_pred=self.rf.predict(self.X_test)

  # Accuracy score
  score=round(accuracy_score(self.y_test,self.y_pred),3)
  print("Accuracy on the Test set: {}".format(score))

  # Classification report
  print(classification_report(self.y_test,self.y_pred))

  # Creating a confusion matrix for training set
  self.y_train_pred=self.rf.predict(self.X_train)

  # Accuracy score
```

```
 score=round(accuracy_score(self.y_train,self.y_train_pred),3)
 print("Accuracy on training set: {}".format(score))

 print(classification_report(self.y_train,self.y_train_pred))

def predict1(self,hemo,rc,sg,al,sc,htn,sod,bp,wc,age):
 print(hemo)
 hemo=float(hemo)
 rc=float(rc)
 sg=float(sg)
 sc=float(sc)
 htn=int(htn)
 sod=float(sod)
 bp=float(bp)
 wc=float(wc)
 age=int(age)

 x=[[hemo,rc,sg,al,sc,htn,sod,bp,wc,age]]
 return self.rf.predict(x)
```

```
 def predictt(self):

self.X_train=self.X_train[['hemo','rc','sg','al','sc','htn','sod','bp','wc','age']]
  self.X_test=self.X_test[['hemo','rc','sg','al','sc','htn','sod','bp','wc','age']]
  self.rf.fit(self.X_train,self.y_train)
  hb3=self.hb2.get()
  rbc3=self.rbc2.get()
  sg3=self.sg2.get()
  al3=self.al2.get()
  sc3=self.sc2.get()
  ht3=self.ht2.get()
  sod3=self.sod2.get()
  bp3=self.bp2.get()
  wbc3=self.wbc2.get()
  age3=self.age2.get()


  prediction =
self.predict1(hb3,rbc3,sg3,al3,sc3,ht3,sod3,bp3,wbc3,age3)[0]
  #prediction =
self.predict1(67.4,7.2,0.99,4,17.0,1,160.6,87,22089,36)[0]
  print(prediction)
  if prediction:
```

```
   print('Oops! You have Chronic Kidney Disease.')
  else:
   print("Great! You don't have Chronic Kidney Disease.")
```

```
if __name__ == '__main__':

 root = Tk()

 application=patient1(root)
 #root.geometry('500x500')
 root.mainloop()
```

# CHAPTER 6

## CONCLUSION AND FUTURE SCOPE FUTURE SCOPE

This would help detect the chances of a person having CKD further on in his life which would be really helpful and cost-effective people. This model could be integrated with normal blood report generation, which could automatically flag out if there is a person at risk. Patients would not have to go to a doctor unless they are flagged by the algorithms. This would• make it cheaper and easier for the modern busy person.

## 6.2CONCLUSION :

This system presented the best prediction algorithm to predict CKD at an early stage. The dataset shows input parameters collected from the CKD patients and the models are trained and validated for the given input parameters. K-Nearest-Neighbors Classifier, Decision Tree Classifier, GaussianNB, Logical Regression and Artificial Neural Network learning models are constructed to carry out the diagnosis of CKD. The performance of the models is evaluated based on a variety of comparison metrics are being used, namely Accuracy, Specificity, Sensitivity and Log Loss. The results of the research showed that Logical Regression model better predicts CKD in comparison to the other models taking all the metrics under consideration. This system would help detect the chances of a person having CKD further on in his life which would be really helpful and cost-effective people. This model could be integrated with normal blood report generation, which could automatically flag out if there is a person at risk. Patients would not have to go to a doctor unless they are flagged by the algorithms. This would make it cheaper and easier for the modern busy person.

# CHAPTER -7

## BIBLIOGRAPHY & REFERENCES

### References Made From:

Operating System Concepts, by Abraham Silberschatz.

M. P. N. M. Wickramasinghe, D. M. Perera and K. A. D. C. P. Kahandawaarachchi, "Dietary prediction for patients with Chronic Kidney Disease (CKD) by considering blood potassium level using machine learning algorithms," 2017 IEEE Life Sciences Conference (LSC), Sydney, NSW, 2017, pp. 300-303

**Sites Referred:**

**http://www.w3schools.com/**

**http://ieeexplore.ieee.org**

http://www.sourcefordgde.com