A project work Submitted in partial fulfilment of

the requirements for the degree of

Bachelor of Computer Science

to the

Bharathidasan University, Tiruchirapalli

By

T.RANJITHA

REG.NO.CB20S202965

N.GAYATHRI

REG.NO.CB20S202950

S.ARUNA

REG.NO.CB20S262230

M.ANISHA

REG.NO.CB20S262228

MIT COLLEGE OF ARTS & SCIENCE FOR WOMAN

MUSIRI -621 211

CONTENT

SYNOPSIS

Chronic kidney disease (CKD) is one of the most critical health problems due to its increasing prevalence. But CKD is a disease which doesn't shows symptoms at all or in some cases it doesn't show any disease specific symptoms it is hard to predict, detect and prevent such a disease and this could be lead to permanently health damage, but machine learning can be hope in this problem it is best in prediction and analysis. Chronic Kidney Disease dataset has been taken from the UCI repository. Implementation of algorithms like naïve bayes, SVM, logistic regression, random forest our proposed method Gradient boosting achieves high accuracy. More than 15 parameters are considered to increase the accuracy of prediction. The performance of the predictions of the algorithms is analyzed using a pre-processed dataset. Instead of detecting whether particular person is affected by CKD or not by analyzing the parameter sugar and Blood pressure we can notify the current stage of patient through recommendation. Hence our system achieves high accuracy in prediction in addition it alert the person to save his/her life.

Many of the people aged 65 or more do have a neurodegenerative disease, which has no cure. If we detect the disease in the early stages, then we can control it. Almost 30%of the patients are facing this incurable disease. Current treatment is available for patients who have minor symptoms.

If these symptoms cannot be found at the early stages , it leads to death.

The main cause for kidney disease is ,

- Diabetes.
- High blood pressure.
- Heart (cardiovascular) disease.
- Smoking.
- Obesity.
- Being Black, Native American or Asian American.
- Family history of kidney disease.

INTRODUCTION

The disability of the kidneys to perform their regular blood filtering function and others is called Chronic Kidney Disease (CKD). The term “chronic” describes the slow degradation of the kidney cells over a long period of time. This disease is a major kidney failure where the kidney sans blood filtering process and there is a heavy fluid buildup in the body. This leads to alarming increase of potasium and calcium salts in the body. Existence of high levels of these salts result in various other ailments in the body. The prime job of kidneys is to filter extra water and wastes from blood. The efficient functioning of this process is important to balance the salts and minerals present in our body. The high balance of salts are necessary to control blood pressure, activate hormones, build red blood cells, etc. A high concentration of calcium leads to various bone diseases and cystic ovaries in women. CKD also may lead to sudden illness or allergy to certain medicines. This state is called as Acute Kidney Injury (AKI). An increased blood pressure may lead to heart problems and heart attacks. CKD in many cases leads to permanent dialysis or kidney transplants. A history of kidney disease in the family also leads to high probability of CKD. Literature shows that almost one out of three people diagnosed with diabetes have CKD. Literature also presents evidences of early identification and care of CKD can improve the quality of the patients life. Prediction algorithms in machine learning can be intelligently used to predict the occurence of CKD and presents a method of early medication.

A person will develop permanent kidney failure. If CKD is not detected and cured in early stage then patient can show following Symptoms: Blood Pressure, anaemia, weekboans, poor nutrition health and nerve damage, Decreased immune response because at advanced stages dangerous levels of fluids, electrolytes, and wastes can build up in your blood and body. Hence it is essential to detect CKD at its early stage but it is unpredictable as its Symptoms develop slowly and aren't specific to the disease. Some people have no symptoms at all so machine learning can be helpful in this problem to predict that the patient has CKD or not. Machine learning does it by using old CKD patient data to train predicting model. Glomerular Filtration Rate (GFR) is the best test to measure your level of kidney function and determine your stage of chronic kidney disease. It can be calculated from the results of your blood creatinine, age, race, gender, and other factors. The earlier disease is detected the better chance of showing or stopping its progression.

Chronic kidney disease (CKD) means your kidneys are damaged and can‘t filter blood the way they should. The disease is called ―chronic‖ because the damage to your kidneys happens slowly over a long period of time. This damage can cause wastes to build up in your body. CKD can also cause other

health problems. Our kidneys have a greater capacity to do their job than is needed to keep us healthy. For example, you can donate one kidney and remain healthy. You can also have kidney damage without any symptoms because, despite the damage, your kidneys are still doing enough work to keep you feeling well. For many people, the only way to know if you have kidney disease is to get your kidneys checked with blood and urine tests.

To predict positive CKD status and the stages of CKD machine learning can be used. Machine Learning grabs a major part of artificial intelligence when it comes to doing predictions from previous data using classification and regression methods. Application of machine learning methods to predict CKD has been explored based on multiple data sets. Among them, the dataset from UCI repository (referred to as UCI dataset hereafter) is identified as a benchmark dataset. Similar to most of the related work, this work considers the mentioned benchmark dataset. When analysing clinical data related to CKD, if there are instances with missing attributes then the missing values handling method should be determined based on the randomness of the way they were missed. Moreover, the UCI data-set has more than 400 instances which are a comparatively small number of samples with 25 attributes.

In this case, the data set may have redundant (highly co-related) features or the data set does not represent all possibilities. Thereby, this work identifies the limitations in handling missing values when analysing CKD data, proposes a new method to handle missing values and presents the evaluation of different methods based on UCI dataset.

KIDNEY DISEASE SYMPTOMS:

MAJOR SYMPTOMS:

- Weight loss and poor appetite.
- Swollen ankles, feet or hands – as a result of water retention (oedema)
- Shortness of breath.
- Tiredness.
- Blood in your pee (urine)
- An increased need to pee – particularly at night.
- Difficulty sleeping (insomnia)
- Itchy skin.

DIAGNOSIS OF PATIENTS WITH CHRONIC KIDNEY DISEASE BY USING TWO FUZZY CLASSIFIERS

The feasibility of two in-house fuzzy classifiers, fuzzy rule-building expert system (FuRES) and fuzzy optimal associative memory (FOAM), for diagnosis of patients with chronic kidney disease (CKD) was investigated.

A linear classifier, partial least squares discriminant analysis (PLS-DA), was used for comparison. The CKD data used in this work were taken from the UCI Machine Learning Repository.

Composite datasets were created by adding different levels of proportional noise to evaluate the robustness of the two fuzzy approaches. Firstly, 11 levels of proportional noises were added to each numeric attribute of the training and prediction sets one after another, and then these simulated training and prediction sets were combined in pairs.

Thus, a grid with 121 groups of simulated data was generated, and classification rates for these 121 pairs were compared. Secondly, the performances of two fuzzy classifiers using the simulated datasets, in which 11 levels of noise were randomly distributed to each numeric attribute, were compared and the average prediction rates of FuRES and FOAM were 98.1±0.5% and 97.2±1.2%, respectively, with 200 bootstrap Latin partitions.

The PLS-DA can give 94.3±0.8% with the identical evaluation. Confluent datasets comprised of the original and modified datasets were also used to evaluate FuRES, FOAM, and PLS-DA classification models.

The average prediction rates of FuRES and FOAM obtained from 200 bootstrapped evaluations were 99.2±0.3% and 99.0±0.3%. PLS-DA yields slightly worse accuracy with 95.9±0.6%.

The results demonstrate that both FuRES and FOAM perform well on the identification of CKD patients, while FuRES is more robust than FOAM. These two fuzzy classifiers are useful tools for the diagnosis of CKD patients with satisfactory robustness, and can also be used for other kinds of patients.

DIAGNOSIS OF CHRONIC KIDNEY DISEASE BY USING RANDOM FOREST

Chronic kidney disease (CKD) is a global public health problem, affecting approximately 10% of the population worldwide. Yet, there is little direct evidence on how CKD can be diagnosed in a systematic and automatic manner. This paper investigates how CKD can be diagnosed by using machine learning (ML) techniques. ML algorithms have been a driving force in detection of abnormalities in different physiological data, and are, with a great success, employed in different classification tasks. In the present study, a number of different ML classifiers are experimentally validated to a real data set, taken from the UCI Machine Learning Repository, and our findings are compared with the findings reported in the recent literature. The results are quantitatively and qualitatively discussed and our findings reveal that the random forest (RF) classifier achieves the near-optimal performances on the identification of CKD subjects. Hence, we show that ML algorithms serve important function in diagnosis of CKD, with satisfactory robustness, and our findings suggest that RF can also be utilized for the diagnosis of similar diseases.

DIAGNOSIS OF CHRONIC KIDNEY DISEASE

As Chronic Kidney Disease progresses slowly, early detection and effective treatment are the only cure to reduce the mortality rate. Machine learning techniques are gaining significance in medical diagnosis because of their classification ability with high accuracy rates. The accuracy of classification algorithms depend on the use of correct feature selection algorithms to reduce the dimension of datasets. In this study, Support Vector Machine classification algorithm was used to diagnose Chronic Kidney Disease. To diagnose the Chronic Kidney Disease, two essential types of feature selection methods namely, wrapper and filter approaches were chosen to reduce the dimension of Chronic Kidney Disease dataset. In wrapper approach, classifier subset evaluator with greedy stepwise search engine and wrapper subset evaluator with the Best First search engine were used.

In filter approach, correlation feature selection subset evaluator with greedy stepwise search engine and filtered subset evaluator with the Best First search engine were used. The results showed that the Support Vector Machine classifier by using filtered subset evaluator with the Best First search engine feature selection method has higher accuracy rate (98.5%) in the diagnosis of Chronic Kidney Disease compared to other selected methods.

END STAGE RENAL DISEASE PATIENTS UNDERGOING DIALYSIS

Chronic Kidney Disease (CKD) anemia is one of the main common comorbidities in patients undergoing End Stage Renal Disease (ESRD).

Iron supplement and especially Erythropoiesis Stimulating Agents (ESA) have become the treatment of choice for that anemia.

However, it is very complicated to find an adequate treatment for every patient in each particular situation since dosage guidelines are based on average behaviors, and thus, they do not take into account the particular response to those drugs by different patients, although that response may vary enormously from one patient to another and even for the same patient in different stages of the anemia.

This work proposes an advance with respect to previous works that have faced this problem using different methodologies (Machine Learning (ML), among others), since the diversity of the CKD population has been explicitly taken into account in order to produce a general and reliable model for the prediction of ESA/Iron therapy response.

Furthermore, the ML model makes use of both human physiology and drug pharmacology to produce a model that outperforms previous approaches, yielding Mean Absolute Errors (MAE) of the Hemoglobin (Hb) prediction around or lower than 0.6 g/dl in the three countries analyzed in the study, namely, Spain, Italy and Portugal.

1.1 ORGANIZING PROFILE

NEXT TECHNOLIGIES is proud to introduce ourselves as an end to training solution providing company for skill development of various sectors.

The training programs of NEXT TECHNOLOGIES encompassa wide range of skills that are integral and necessary parts of today's competitive global business atmosphere.

OUR MOTTO

Discover your potential!! Realize your dreams!!

OUR VISION

Our vision is to emphasize high priority in providing training solutions to meet the demand of the ever growing skilled manpower of global standards along with high moral, ethical, social values of our country.

OUR MISSION

Our mission is to accelerate economic development in India, by creating employment opportunities, and making our youth employable, we believe we can impact large scale economic growth.

OUR TEAM

Our trainers are a team of experienced, passionate, and realistic with strong academic background. We formally build a team who can live it to the best of it and for us commercial part is always a secondary note.

OUR EXPERTISE

NEXT TECHNOLOGIES Trainers team served across industries and levels. Our familiarity and confidence in this space has given us the ability to maintain the highest standards of quality to meet the objectives of our clients.

1.2 SYSTEM REQUIREMENTS

1.2.1 HARDWARE REQUIREMENTS (Minimum requirement):

The section of hardware configuration is an important task related to the software development insufficient random access memory may affect adversely on the speed and efficiency of the entire system. The process should be powerful to handle the entire operations. The hard disk should have sufficient capacity to store the file and application.

- System : Pentium Dual Core.
- Hard Disk : 120 GB.
- Ram : 1GB.

1.2.2 SOFTWARE REQUIREMENTS:

A major element in building a system is the section of compatible software since the software in the market is experiencing in geometric progression. Selected software should be acceptable by the firm and one user as well as it should be feasible for the system.

This document gives a detailed description of the software requirement specification. The study of requirement specification is focused specially on the functioning of the system. It allow the developer or analyst to understand the system, function to be carried out the performance level to be obtained and corresponding interfaces to be established.

- Operating system : Windows 7.
- Coding Language : Python
- Database : MYSQL

1.2.3 SOFTWARE DESCRIPTION :

PYTHON

In technical terms, Python is an object-oriented, high-level programming language with integrated dynamic semantics primarily for web and app development. It is extremely attractive in the field of Rapid Application Development because it offers dynamic typing and dynamic binding options.

Python is relatively simple, so it's easy to learn since it requires a unique syntax that focuses on readability. Developers can read and translate Python code much easier than other languages. In turn, this reduces the cost of program maintenance and development because it allows teams to work collaboratively without significant language and experience barriers.

Additionally, Python supports the use of modules and packages, which means that programs can be designed in a modular style and code can be reused across a variety of projects. Once you've developed a module or package you need, it can be scaled for use in other projects, and it's easy to import or export these modules.

One of the most promising benefits of Python is that both the standard library and the interpreter are available free of charge, in both binary and source form. There is no exclusivity either, as Python and all the necessary tools are available on all major platforms. Therefore, it is an enticing option for developers who don't want to worry about paying high development costs.
If this description of Python over your head, don't worry. You'll understand it soon enough. What you need to take away from this section is that Python is a programming language used to develop software on the web and in app form, including mobile. It's relatively easy to learn, and the necessary tools are available to all free of charge.

That makes Python accessible to almost anyone. If you have the time to learn, you can create some amazing things with the language.

READABLE AND MAINTAINABLE CODE

While writing a software application, you must focus on the quality of its source code to simplify maintenance and updates. The syntax rules of Python allow you to express concepts without writing additional code. At the same time, Python, unlike other programming languages, emphasizes on code readability, and allows you to use English keywords instead of punctuations. Hence, you can use Python to build custom applications without writing additional code. The readable and clean code base will help you to maintain and update the software without putting extra time and effort.

MULTIPLE PROGRAMMING PARADIGMS :

Like other modern programming languages, Python also supports several programming paradigm. It supports object oriented and structured programming fully. Also, its language features support various concepts in functional and aspect-oriented programming. At the same time, Python also features a dynamic type system and automatic memory management. The programming paradigms and language features help you to use Python for developing large and complex software applications.

COMPATIBLE WITH MAJOR PLATFORMS AND SYSTEMS :

At present, Python is supports many operating systems. You can even use Python interpreters to run the code on specific platforms and tools. Also, Python is an interpreted programming language. It allows you to you to run the same code on multiple platforms without recompilation. Hence, you are not required to recompile the code after making any alteration. You can run the modified application code without recompiling and check the impact of changes made to the code immediately. The feature makes it easier for you to make changes to the code without increasing development time.

ROBUST STANDARD LIBRARY :

Its large and robust standard library makes Python score over other programming languages. The standard library allows you to choose from a wide range of modules according to your precise needs. Each module further enables you to add functionality to the Python application without writing additional code. For instance, while writing a web application in Python, you can use specific modules to implement web services, perform string operations, manage operating system interface or work with internet protocols. You can even gather information about various modules by browsing through the Python Standard Library documentation.

MANY OPEN SOURCE FRAMEWORKS AND TOOLS :

As an open source programming language, Python helps you to curtail software development cost significantly. You can even use several open source Python frameworks, libraries and development tools to curtail development time without increasing development cost. You even have option to choose from a wide range of open source Python frameworks and development tools according to your precise needs. For instance, you can simplify and speedup web application development by using robust Python web frameworks like Django, Flask, Pyramid, Bottle and Cherrypy. Likewise, you can accelerate desktop GUI application development using Python GUI frameworks and toolkits like PyQT, PyJs, PyGUI, Kivy, PyGTK and WxPython.

SIMPLIFY COMPLEX SOFTWARE DEVELOPMENT :

Python is a general purpose programming language. Hence, you can use the programming language for developing both desktop and web applications. Also, you can use Python for developing complex scientific and numeric applications. Python is designed with features to facilitate data analysis and visualization. You can take advantage of the data analysis features of Python to create custom big data solutions without putting extra time and effort.

At the same time, the data visualization libraries and APIs provided by Python help you to visualize and present data in a more appealing and effective way. Many Python developers even use Python to accomplish artificial intelligence (AI) and natural language processing tasks.

ADOPT TEST DRIVEN DEVELOPMENT :

We can use Python to create prototype of the software application rapidly. Also, you can build the software application directly from the prototype simply by refactoring the Python code. Python even makes it easier for you to perform coding and testing simultaneously by adopting test driven development (TDD) approach. You can easily write the required tests before writing code and use the tests to assess the application code continuously. The tests can also be used for checking if the application meets predefined requirements based on its source code.

However, Python, like other programming languages, has its own shortcomings. It lacks some of the built-in features provided by other modern programming language. Hence, you have to use Python libraries, modules, and frameworks to accelerate custom software development. Also, several studies have shown that Python is slower than several widely used programming languages including Java and C++. You have to speed up the Python application by making changes to the application code or using custom runtime. But you can always use Python to speed up software development and simplify software maintenance.

BENEFITS OF LEARNING PYTHON :

There are many benefits of learning Python, especially as your first language, which we will discuss.

It is a language that is remarkably easy to learn, and it can be used as a stepping stone into other programming languages and frameworks. If you're an absolute beginner and this is your first time working with any type of coding language, that's something you definitely want.

Python is widely used, including by a number of big companies like Google, Pinterest, Instagram, Disney, Yahoo!, Nokia, IBM, and many others.

The Raspberry Pi - which is a mini computer and DIY lover's dream - relies on Python as it's main programming language too. You're probably wondering why either of these things matter, and that's because once you learn Python, you'll never have a shortage of ways to utilize the skill. Not to mention, since a lot of big companies rely on the language, you can make good money as a Python developer.

OTHER BENEFITS INCLUDE:

1) Python can be used to develop prototypes, and quickly because it is so easy to work with and read.

2) Most automation, data mining, and big data platforms rely on Python. This is because it is the ideal language to work with for general purpose tasks.

3) Python allows for a more productive coding environment than massive languages like C# and Java. Experienced coders tend to stay more organized and productive when working with Python, as well.

4) Python is easy to read, even if you're not a skilled programmer. Anyone can begin working with the language, all it takes is a bit of patience and a lot of practice. Plus, this makes it an ideal candidate for use among multi-programmer and large development teams.

5) Python powers Django, a complete and open source web application framework. Frameworks - like Ruby on Rails - can be used to simplify the development process.

6) It has a massive support base thanks to the fact that it is open source and community developed. Millions of like-minded developers work with the language on a daily basis and continue to improve core functionality.

The latest version of Python continues to receive enhancements and updates as time progresses. This is a great way to network with other developers.

PYTHON ENVIRONMENT SETUP :

One of the most important things you'll do when working with any programming language is setup a development environment which allows you to execute the code you write. Without this, you will never be able to check your work and see if your website or application is free of syntax errors.

With Python, you also need something called an interpreter that converts your code - which makes up the entirety of your application - to something the computer can read and execute. Without this interpreter, you'll have no way to run your code.

To convert your code, you must first use a Python shell, which calls upon the interpreter through something called a "bang" line.

As for creating an application or file, there are two ways to do this. You can create a program using a simple text editor like WordPad, or Notepad++. You can also create a program using a Python shell. There are advantages and disadvantages to each method, which we'll discuss next.

PYTHON SHELL VERSUS TEXT FILE :

A shell is a program or tool that can be used to interact with a system. For instance, the Windows operating system shell can be tapped into by using a "terminal" or command line to submit commands and arguments.

With Python, things work a bit differently than an operating system shell. The Python shell is used to interact with an interpreter, which feeds code to a computer in a form that it can understand.

When you execute a Python program that you've written, the interpreter reads the code and converts it into usable commands. The important thing to note is that all of this is done after the program has been executed.

With a shell, the interpreting - or conversion - happens in real-time as you type the code into the computer or system. This means that the actual program is executing as you type. This gives you some idea of how your final code will look, and what your program is actually going to do.

When you write code in a text file, none of that happens until you feed the document into an interpreter. If you have Python installed on your computer you can call upon the interpreter using a command line, but this step is done after you've already written the code.

This makes it more difficult to spot errors in your code, and it can also be frustrating if the interpreter runs into issues, because they may not be as apparent as they would if you had used a shell. Still, a lot of developers prefer to use a text editing tool because it is simple and easy to do.

PYTHON FEATURES:

Python is often comparable to Perl, Ruby, PHP, Scheme, and Java. This is because it is is an incredibly powerful object-oriented language.

Python also has several notable features which make it an enticing language to work with for developers.

1) Python makes use of an elegant syntax, meaning the programs you write are much easier to read. This is because they are closer to the human language, or how we write our words, instead of a language that computers use to read and interpret code.

   For example, the "print" command will display anything proceeding it - and in quotes - at runtime.

2) Python is simple and easy-to-use, which means that it's much easier to get your programs up and running. That is why Python is considered ideal for prototype development and similar ad-hoc programming tasks. It does not compromise maintainability either.

2.SYSTEM STUDY

FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- ECONOMICAL FEASIBILITY
- TECHNICAL FEASIBILITY

ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client.

The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

2.1 EXISTING SYSTEM:

2.1.1 DESCRIPTION

In healthcare organization, Classification is one in all the foremost usually used ways of machine learning. The classification model shows the class of result for each data point. The classifying methods are Logistic regression, Support Vector Machine and K-Nearest Neighbor. KNN is used to visualize at the relationship between different CKD risk factors, in order to predict the disease at an early stage.

Machine Learning is a growing phase dealing with the study of a huge variable data and it is grown from the study of pattern (speech and handwriting) recognition and computational learning theory in Artificial Intelligence having numerous methods, algorithms, and techniques to analyze and predict the data.

Machine Learning techniques have proved huge success in detection and recognition of many essential diseases in medical science's point of view.

Machine learning would thus be useful for predicting whether the patient has CKD or not in this question. By using old CKD patient data to train predictive model, Machine Learning does so. Many researches are implementing different machine learning algorithms to identify accuracy level of prediction. In existing system also the prediction of particular patient as normal or abnormal is predicted with different accuracy rate.

2.1.2 DRAWBACKS :

- Accuracy level of prediction increases and need to improve.
- No intimation or alert message is notified.
- Minimum number of dataset is used for processing should be improved to increase accuracy of prediction.

2.2 PROPOSED SYSTEM:

2.2.1 DESCRIPTION

CKD, in its early stages, has no symptoms; testing may be the only way to find out if the patient has kidney disease. Early detection of CKD in its initial stages can help the patient get effective treatment and then prohibit the progression to end-stage renal disease (ESRD). Machine learning is an application of AI that gives systems the power to automatically learn and improve from experience without being explicitly programmed.

Initially dataset gathered are preprocessed, feature extracted and classification are done to train dataset. In our proposed method Gradient Boosting is a supervised machine learning ensemble process used for regression and classification problems.

The model is built stage wise and it is generalized by allowing them to optimize a differentiable loss function.

Whether the patient is affected by CKD or not two parameters such as Sugar level and blood pressure is considered and verified with normal value of a human being if particular value is normal it will be intimated as normal or it is nearer to limit value then it will be notified to particular person as a alert message. Hence it create an alert to take precaution steps to have a healthy life.

ADVANTAGES:

- Accuracy of prediction is increased.
- Notification of patient level is intimated based on 2 parameters.
- More number of dataset is included in processing increases accuracy level in prediction.

3.SYSTEM DESIGN AND DEVELOPMENT

3.1 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.

3.2 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.

3.3 DATABASE DESIGN

The Database Management System (DBMS) consists of a collection of interrelated data and a set of programs to access that data. The Collection of data usually referred to as database. The primary key goal of DBMS is to provide an environment that is both convenient and efficient to use in retrieving and storing data information.

CODE DESIGN:

A code design is a document that sets rules for the design of a new development. It is a tool that can be used in the design and planning process, but goes further and is more regulatory than other forms of guidance commonly used in the English planning system over recent decades.

It can be thought of as a process and document – and therefore a mechanism – which operationalises design guidelines or standards which have been established through a master plan process.

The master plan or framework is the vision.

It should be accompanied by a design rationale that explains why, followed by a code that gives instructions to the appropriate degree or precision and that is operational.

In this way a design code may be a tool which helps ensure that the aspirations for quality and quantity for housing developments, particularly for large-scale projects, sought by the Government and other agencies are actually realised in the final schemes.

3.4. SYSTEM DEVELOPMENT

3.4.1 DESCRIPTION OF MODULES:

1. Dataset gathering
2. Preprocessing of data
3. Feature selection
4. Classification algorithm
5. Intimation module

DATASET:

Chronic Kidney Disease dataset is used for this research work. Many researchers had also used this dataset. This dataset is being provided by the UC Irvine Machine Learning Repository and it is available on the UCI website.

This dataset contains more than 400 instances and 24 attributes with 1 target attribute. The target attribute has labelled in two-class to represent CKD or non-CKD. The dataset was collected from various hospitals it also contains missing value.

PREPROCESSING OF DATA:

Data preprocessing could be a strategy that is utilized to change over the raw information into a clean dataset.

It is a basic step to train every machine learning classifier algorithm. This technique concludes such actions as handle missing values, rescaling of the dataset, transform into binary data and standardize of the dataset.

When the dataset included attributes with varying scales, rescaling is used to scale the dataset.

The binary transformation has been applied to convert the value into 0 and 1. All values of every attribute are considered as 1 for above the threshold and as 0 for below the threshold.

Standardized method ensures that each attribute has mean 0 and standard deviation 1.

FEATURE SELECTION:

Feature selection is needed for trained each machine learning classifier because without removing unnecessary attributes from the dataset result may be affected.

The classifier algorithm with feature selection gives better performance and reduces the execution time of the model. For this process, three different feature selection methods were used in this research.

FILTER METHOD:

The filter is one of the methods to select the appropriate feature. It selects the feature on their integral features without integrating any learning classifier algorithm. This method gives result faster as compared to the wrapper method.

The method assigns the score to every attribute based on their statistical correlator between attributes. There are many filter methods are available, but Correlation-based Feature Selection (CFS) method has been used. CFS is the algorithm to select the feature-based on the attribute ranks.

It assigns the rank to attribute subset as based on the correlation heuristic evaluation function.

The function works on the strategy that creates two class labels, one is correlated to class and low correlated class and selects only correlated label class attributes.

CLASSIFICATION ALGORITHMS:

Classification technique is an important feature of supervised learning. Classifiers learn from the training dataset and apply on the testing dataset for finding the target attribute.Gradient boosting (GB) is an ensemble boosting technique that starts with “regression tree” as “weak learners”.

In general, the GB model adds an additive model to minimize the loss function by using a stage-wise sampling strategy. The loss function measures the amount at which the expected value deviates from the real value.

Stage wise fashion put more emphasis on samples that are difficult to predict or misclassified. Unlike random forest, in GB, samples that are misclassified have a higher chance of being selected in training data.

GB reduces bias and variance and often provides higher accuracy, but the parameters should be tuned carefully to avoid overfitting. Therefore, nested cross-validation has been applied.

INTIMATION MODULE:

In addition to the accuracy prediction of CKD our proposed method intimate alert message to patient based on the parameter sugar level and blood pressure.

Normal range of these two parameters will be included in code and from the inputted patient detail these two parameter value will be extracted and verified with normal range value.

4. TESTING AND IMPLEMENTATION

In scikit-learn python library, from sklearn.linear_model importLogisticRegression Module is used for carrying out the Logistic Regression. We have to specify the iterations to the function parameter and assign an object to the classifier. We will use our training dataset to fit the model. Fig 3.15 shows the sample code for trainingmodel using Logistic Regression.

TESTING DATA

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product.

It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

TYPES OF TESTS

UNIT TESTING

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration.

This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

INTEGRATION TESTING

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

FUNCTIONAL TEST

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input	:	identified classes of valid input must be accepted.
Invalid Input	:	identified classes of invalid input must be rejected.
Functions	:	identified functions must be exercised.
Output	:	identified classes of application outputs must be exercised.
Systems/Procedures	:	interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing.

Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

SYSTEM TEST

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results.

An example of system testing is the configuration oriented system integration test.

System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

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.

UNIT TESTING :

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

TEST STRATEGY AND APPROACH

Field testing will be performed manually and functional tests will be written in detail.

TEST OBJECTIVES

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

FEATURES TO BE TESTED

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

INTEGRATION TESTING

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

TEST RESULTS : All the test cases mentioned above passed successfully. No defects encountered.

ACCEPTANCE TESTING

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

TEST RESULTS : All the test cases mentioned above passed successfully. No defects encountered.

Once Kidney disease Prediction model has been trained on the pre-processed dataset, then the model is tested using different data points. In this testing step, the model is checked for correctness and accuracy by providing a test dataset to it. All the training methods need to be verified for finding out the best model to be used. In figure 3.15, after fitting our model with training data, we used this model to predict values for the test dataset. These predicted values on testing data are used for model comparison and accurate calculation.

```
from sklearn.linear_model import LogisticRegression
lr=LogisticRegression(random_state=0,max_iter=300)
lr.fit(x_train,y_train)
y_head=lr.predict(x_test)
print("Logistic Regression testaccuracy ",lr.score(x_test,y_test))
```

```
Logistic Regression testaccuracy  0.7929515418502202
```

```
classid,tn,fp,fn,tp=perf_measure(y_test,y_head)
auc_scor.append(roc_auc_score(y_test,y_head))
score_list.append(accuracy(classid,tn,fp,fn,tp))
precision_scor.append(precision(classid,tn,fp,fn,tp))
recall_scor.append(recall(classid,tn,fp,fn,tp))
f1_scor.append(f1_score(y_test,y_head,average='macro'))
NPV_scor.append(NPV(classid,tn,fp,fn,tp))
specificity_scor.append(specificity(classid,tn,fp,fn,tp))
```

```
Logistic Regression report:
              precision    recall  f1-score   support

           0       0.54      0.15      0.23        48
           1       0.81      0.97      0.88       179

    accuracy                           0.79       227
   macro avg       0.67      0.56      0.55       227
weighted avg       0.75      0.79      0.74       227
```

Fig-3.15 Logistic Regression model

5.CONCLUSION

In the context of developing countries, the costs resulting from the usage of software to assist in CKD diagnoses needs to be as low as possible, especially in hard-to-reach and rural settings. The number of CKD attributes used during CKD risk classifications impacts the cost of usage and the performance of the classifiers.

The machine learning techniques present different levels of accuracy for the CKD diagnosis depending on the number of attributes considered during the classification. In this study, the gradient boosting exhibited the best performance using the CKD dataset, comprising of hypertension, DM, creatinine, urea, albuminuria, age, gender, and GFR attributes.

These attributes are commonly used by nephrologists to diagnose CKD in developing countries. Nevertheless, the RF machine learning technique usually conducts more complex evaluations, making the interpretation of the classification results by physicians difficult. The application of algorithms (interpreters) is required to interpret the results before presenting them to primary care physicians. In addition, critical misleading classifications are not presented by the gradient boosting when evaluating the CKD dataset subjects.

6.BIBLOGRAPHY & REFERENCES

1. Zewei Chen, Zhuoyong Zhang, Ruohua Zhu and Yuhong Xiang, “Diagnosis of patients with chronic kidney disease by using two fuzzy classifiers” March 2016Chemometrics and Intelligent Laboratory Systems 153.

2. Abdulhamit Subasi, E mina AlickovicJasmin Kevric, “Diagnosis of Chronic Kidney Disease by Using Random Forest” Part of the IFMBE Proceedings book series (IFMBE, volume 62) 15 March 2017.

3. Huseyin Polat, Homay Danaei Mehr, Aydin Cetin, “Diagnosis of Chronic Kidney Disease Based on Support Vector Machine by Feature Selection Methods” National library of medicine J Med Syst . 2017 Apr;41(4):55. Epub 2017 Feb 27.

4. Carlo Barbieri, Flavio Mari, Andrea Stopper, Emanuele Gatti, Pablo Escandell-Montero, José M Martínez-Martínez, José D Martín-Guerrero, “A new machine learning approach for predicting the response to anemia treatment in a large cohort of End Stage Renal Disease patients undergoing dialysis” Comput Biol Med . 2015 Jun;61:56-61. doi: 10.1016/j.compbiomed.2015.03.019. Epub 2015 Mar 23.

5. Nathan R. Hill, Samuel T. Fatoba, Jason L. Oke, Jennifer A. Hirst, Christopher A. O’Callaghan, Daniel S. Lasserson, F. D. Richard Hobbs, “Global Prevalence of Chronic Kidney Disease – A Systematic Review and Meta-Analysis” Plos one Published: July 6, 2016.

6. Mohamed Alloghani, Dhiya Al-Jumeily Obe, Thar Baker and Abir Hussain, “Applications of Machine Learning Techniques for Software Engineering Learning and Early Prediction of Students” Performance: 4th International Conference, SCDS 2018, Bangkok, Thailand, August 15-16, 2018, Proceedings.

7. Deepa Gupta; Sangita Khare; Ashish Aggarwal, “A method to predict diagnostic codes for chronic diseases using machine learning techniques” 2016 International Conference on Computing, Communication and Automation (ICCCA).

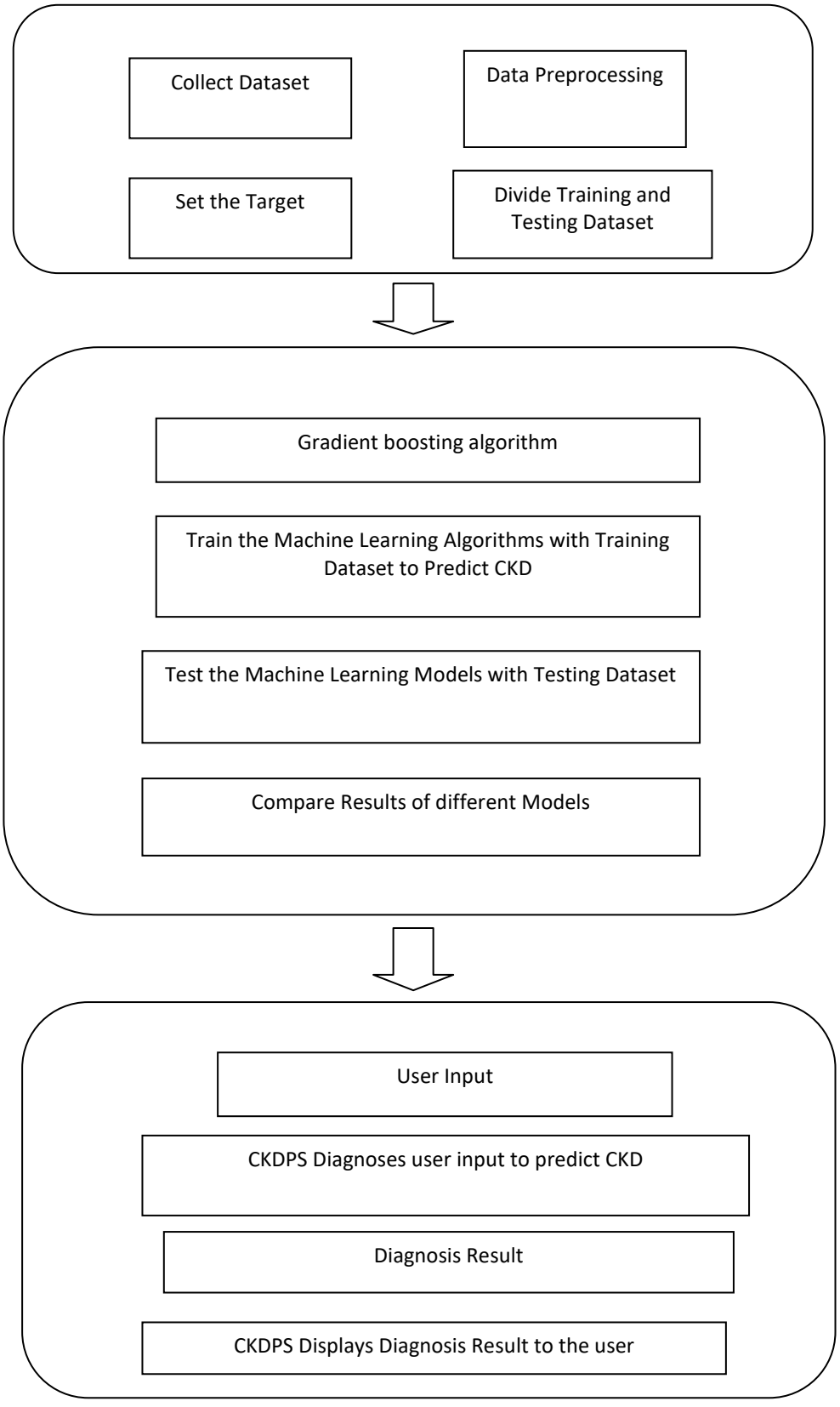
8. S.Ramya, Dr.N.Radha, "Diagnosis of Chronic Kidney Disease Using Machine Learning Algorithms," Proc. Internat ional Journal of Innovative Research in Computer and Communication Engineering, Vol. 4, Issue 1, January 2016.

9. S. A. Shinde and P. R. Rajeswari, “ Intelligent health risk predict ion systems using machine learning: a review,” IJET, vol. 7, no. 3, pp. 1019– 1023, 2018.

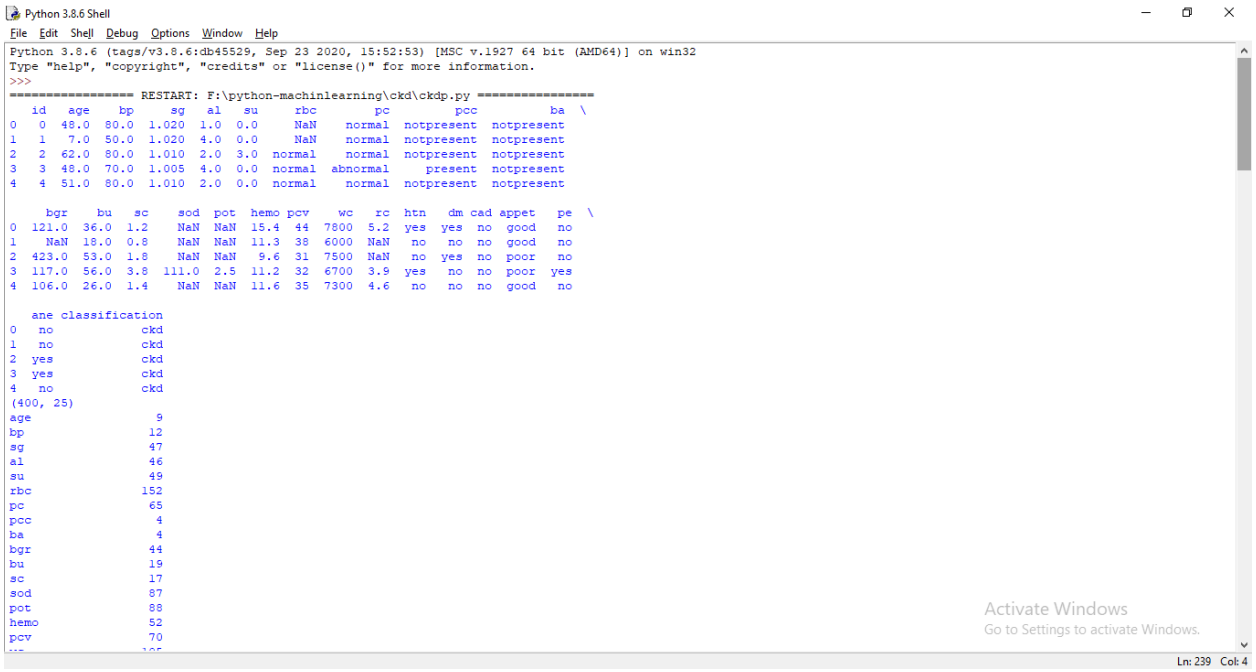
10. A.J. Aljaaf et al, "Early predict ion of chronic renal disorder mistreatment machine learning supported by prognost icat ive analytics," in 2018 IEEE Congress on organic process Computat ion (CEC), 2018.

7.APPENDICES

A.DATA FLOW DIAGRAM



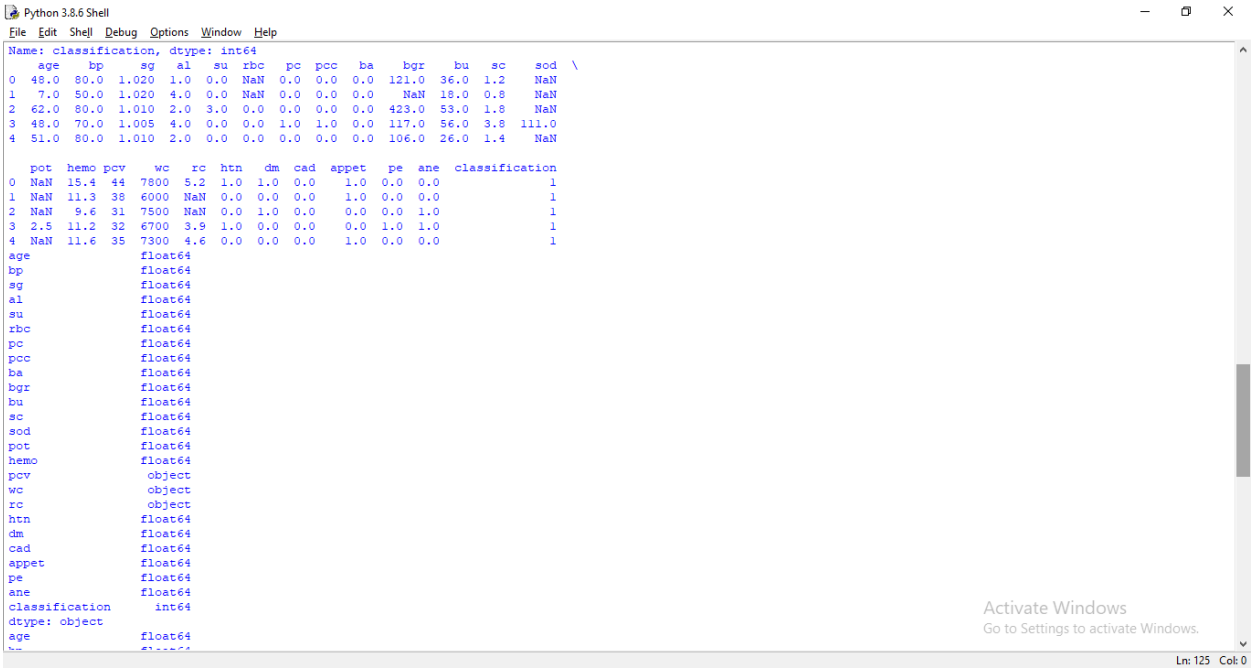
B. DATASET :

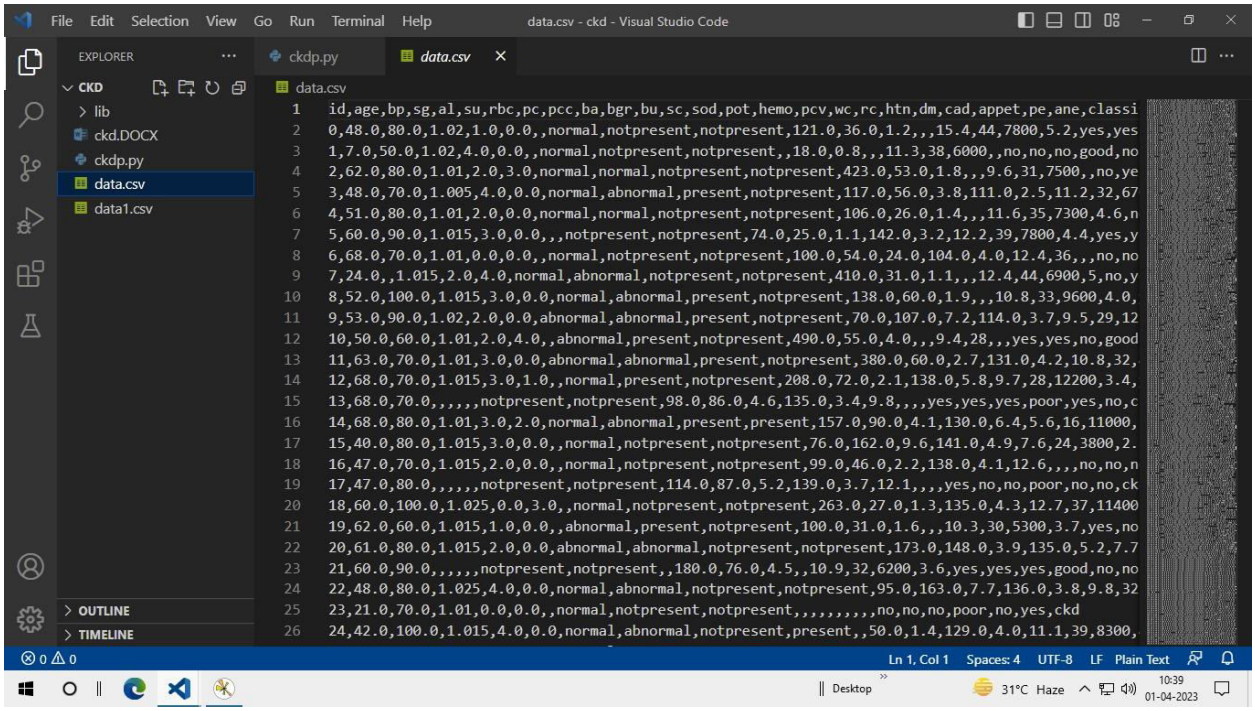


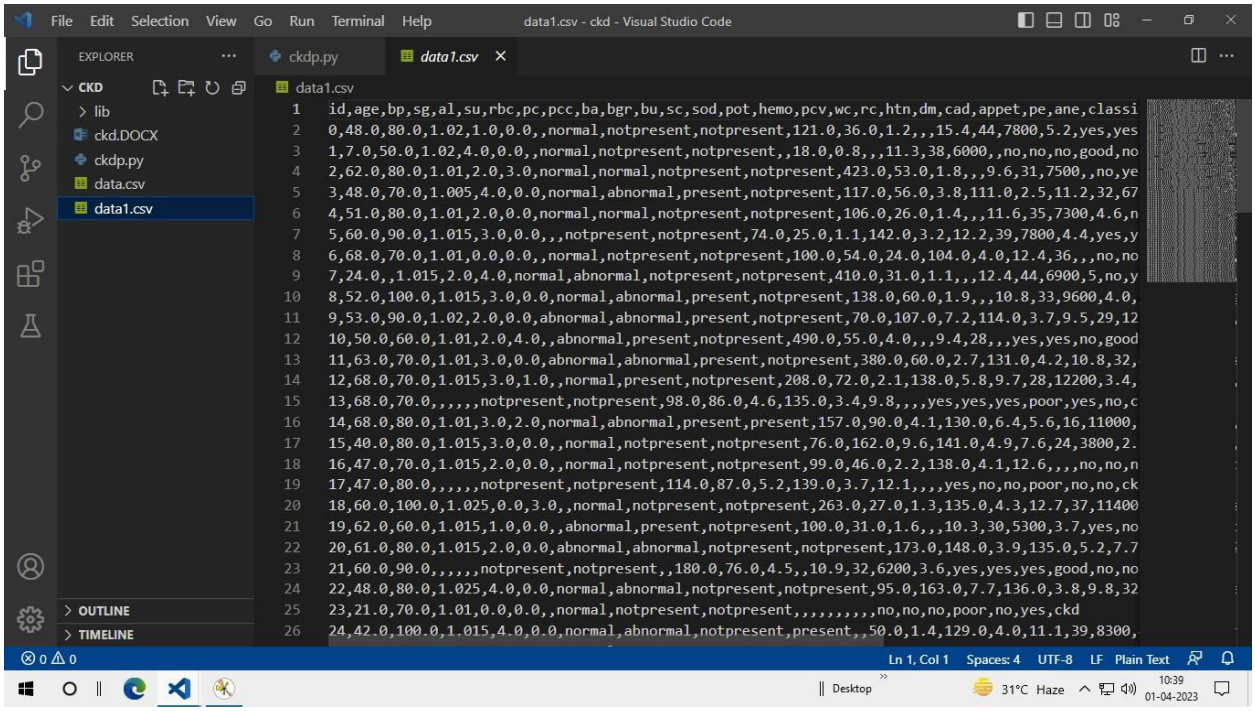
```
Python 3.8.6 Shell
File  Edit  Shell  Debug  Options  Window  Help
Python 3.8.6 (tags/v3.8.6:db45529, Sep 23 2020, 15:52:53) [MSC v.1927 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: F:\python-machinlearning\ckd\ckdp.py ==========
   id   age    bp     sg   al   su     rbc        pc         pcc          ba  \
0   0  48.0  80.0  1.020  1.0  0.0     NaN    normal  notpresent  notpresent
1   1   7.0  50.0  1.020  4.0  0.0     NaN    normal  notpresent  notpresent
2   2  62.0  80.0  1.010  2.0  3.0  normal    normal  notpresent  notpresent
3   3  48.0  70.0  1.005  4.0  0.0  normal  abnormal     present  notpresent
4   4  51.0  80.0  1.010  2.0  0.0  normal    normal  notpresent  notpresent

     bgr     bu   sc     sod  pot  hemo pcv    wc   rc  htn   dm cad appet   pe  \
0  121.0   36.0  1.2     NaN  NaN  15.4  44  7800  5.2  yes  yes  no  good   no
1    NaN   18.0  0.8     NaN  NaN  11.3  38  6000  NaN   no   no  no  good   no
2  423.0   53.0  1.8     NaN  NaN   9.6  31  7500  NaN   no  yes  no  poor   no
3  117.0   56.0  3.8   111.0  2.5  11.2  32  6700  3.9  yes   no  no  poor  yes
4  106.0   26.0  1.4     NaN  NaN  11.6  35  7300  4.6   no   no  no  good   no

   ane classification
0   no            ckd
1   no            ckd
2  yes            ckd
3  yes            ckd
4   no            ckd
(400, 25)
age                9
bp                12
sg                47
al                46
su                49
rbc              152
pc                65
pcc                4
ba                 4
bgr               44
bu                19
sc                17
sod               87
pot               88
hemo              52
pcv               70
                 ...
```


```
Python 3.8.6 Shell
File  Edit  Shell  Debug  Options  Window  Help
htn                2
dm                 2
cad                2
appet              1
pe                 1
ane                1
classification     0
dtype: int64
              age          bp          sg          al          su          bgr  \
count  391.000000  388.000000  353.000000  354.000000  351.000000  356.000000
mean    51.483376   76.469072    1.017408    1.016949    0.450142  148.036517
std     17.169714   13.683637    0.005717    1.352679    1.099191   79.281714
min      2.000000   50.000000    1.005000    0.000000    0.000000   22.000000
25%     42.000000   70.000000    1.010000    0.000000    0.000000   99.000000
50%     55.000000   80.000000    1.020000    0.000000    0.000000  121.000000
75%     64.500000   80.000000    1.020000    2.000000    0.000000  163.000000
max     90.000000  180.000000    1.025000    5.000000    5.000000  490.000000

               bu          sc         sod         pot        hemo
count  381.000000  383.000000  313.000000  312.000000  348.000000
mean    57.425722    3.072454  137.528754    4.627244   12.526437
std     50.503006    5.741126   10.408752    3.193904    2.912587
min      1.500000    0.400000    4.500000    2.500000    3.100000
25%     27.000000    0.900000  135.000000    3.800000   10.300000
50%     42.000000    1.300000  138.000000    4.400000   12.650000
75%     66.000000    2.800000  142.000000    4.900000   15.000000
max    391.000000   76.000000  163.000000   47.000000   17.800000
age              float64
bp               float64
sg               float64
al               float64
su               float64
rbc               object
pc                object
pcc               object
ba                object
bgr              float64
bu               float64
sc               float64
sod              float64
pot              float64
hemo             float64
```


Python 3.8.6 Shell
File Edit Shell Debug Options Window Help
ane object
classification object
dtype: object
normal 201
abnormal 47
Name: rbc, dtype: int64
normal 259
abnormal 76
Name: pc, dtype: int64
notpresent 354
present 42
Name: pcc, dtype: int64
notpresent 374
present 22
Name: ba, dtype: int64
no 251
yes 147
Name: htn, dtype: int64
no 258
yes 134
\tno 3
\tyes 2
yes 1
Name: dm, dtype: int64
no 362
yes 34
\tno 2
Name: cad, dtype: int64
['good' 'poor' nan]
no 323
yes 76
Name: pe, dtype: int64
no 339
yes 60
Name: ane, dtype: int64
ckd 248
notckd 150
ckd\t 2
Name: classification, dtype: int64
age bp sg al su rbc pc pcc ba bgr bu sc sod \
0 48.0 80.0 1.020 1.0 0.0 NaN 0.0 0.0 0.0 121.0 36.0 1.2 NaN
Activate Windows
Go to Settings to activate Windows.
Ln: 83 Col: 0


Python 3.8.6 Shell
File Edit Shell Debug Options Window Help
Name: classification, dtype: int64
age bp sg al su rbc pc pcc ba bgr bu sc sod \
0 48.0 80.0 1.020 1.0 0.0 NaN 0.0 0.0 0.0 121.0 36.0 1.2 NaN
1 7.0 50.0 1.020 4.0 0.0 NaN 0.0 0.0 0.0 NaN 18.0 0.8 NaN
2 62.0 80.0 1.010 2.0 3.0 0.0 0.0 0.0 0.0 423.0 53.0 1.8 NaN
3 48.0 70.0 1.005 4.0 0.0 0.0 1.0 1.0 0.0 117.0 56.0 3.8 111.0
4 51.0 80.0 1.010 2.0 0.0 0.0 0.0 0.0 0.0 106.0 26.0 1.4 NaN
pot hemo pcv wc rc htn dm cad appet pe ane classification
0 NaN 15.4 44 7800 5.2 1.0 1.0 0.0 1.0 0.0 0.0 1
1 NaN 11.3 38 6000 NaN 0.0 0.0 0.0 1.0 0.0 0.0 1
2 NaN 9.6 31 7500 NaN 0.0 1.0 0.0 0.0 0.0 1.0 1
3 2.5 11.2 32 6700 3.9 1.0 0.0 0.0 0.0 1.0 1.0 1
4 NaN 11.6 35 7300 4.6 0.0 0.0 0.0 1.0 0.0 0.0 1
age float64
bp float64
sg float64
al float64
su float64
rbc float64
pc float64
pcc float64
ba float64
bgr float64
bu float64
sc float64
sod float64
pot float64
hemo float64
pcv object
wc object
rc object
htn float64
dm float64
cad float64
appet float64
pe float64
ane float64
classification int64
dtype: object
age float64
Activate Windows
Go to Settings to activate Windows.
Ln: 125 Col: 0


File Edit Selection View Go Run Terminal Help
data.csv - ckd - Visual Studio Code
EXPLORER
ckdp.py
data.csv
CKD
lib
ckd.DOCX
ckdp.py
data.csv
data1.csv
OUTLINE
TIMELINE
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,classi
0,48.0,80.0,1.02,1.0,0.0,,normal,notpresent,notpresent,121.0,36.0,1.2,,,15.4,44,7800,5.2,yes,yes
1,7.0,50.0,1.02,4.0,0.0,,normal,notpresent,notpresent,,18.0,0.8,,,11.3,38,6000,,no,no,no,good,no
2,62.0,80.0,1.01,2.0,3.0,normal,normal,notpresent,notpresent,423.0,53.0,1.8,,,9.6,31,7500,,no,ye
3,48.0,70.0,1.005,4.0,0.0,normal,abnormal,present,notpresent,117.0,56.0,3.8,111.0,2.5,11.2,32,67
4,51.0,80.0,1.01,2.0,0.0,normal,normal,notpresent,notpresent,106.0,26.0,1.4,,,11.6,35,7300,4.6,n
5,60.0,90.0,1.015,3.0,0.0,,,notpresent,notpresent,74.0,25.0,1.1,142.0,3.2,12.2,39,7800,4.4,yes,y
6,68.0,70.0,1.01,0.0,0.0,,normal,notpresent,notpresent,100.0,54.0,24.0,104.0,4.0,12.4,36,,,no,no
7,24.0,,1.015,2.0,4.0,normal,abnormal,notpresent,notpresent,410.0,31.0,1.1,,,12.4,44,6900,5,no,y
8,52.0,100.0,1.015,3.0,0.0,normal,abnormal,present,notpresent,138.0,60.0,1.9,,,10.8,33,9600,4.0,
9,53.0,90.0,1.02,2.0,0.0,abnormal,abnormal,present,notpresent,70.0,107.0,7.2,114.0,3.7,9.5,29,12
10,50.0,60.0,1.01,2.0,4.0,,abnormal,present,notpresent,490.0,55.0,4.0,,,9.4,28,,,yes,yes,no,good
11,63.0,70.0,1.01,3.0,0.0,abnormal,abnormal,present,notpresent,380.0,60.0,2.7,131.0,4.2,10.8,32,
12,68.0,70.0,1.015,3.0,1.0,,normal,present,notpresent,208.0,72.0,2.1,138.0,5.8,9.7,28,12200,3.4,
13,68.0,70.0,,,,,notpresent,notpresent,98.0,86.0,4.6,135.0,3.4,9.8,,,,yes,yes,yes,poor,yes,no,c
14,68.0,80.0,1.01,3.0,2.0,normal,abnormal,present,present,157.0,90.0,4.1,130.0,6.4,5.6,16,11000,
15,40.0,80.0,1.015,3.0,0.0,,normal,notpresent,notpresent,76.0,162.0,9.6,141.0,4.9,7.6,24,3800,2.
16,47.0,70.0,1.015,2.0,0.0,,normal,notpresent,notpresent,99.0,46.0,2.2,138.0,4.1,12.6,,,,no,no,n
17,47.0,80.0,,,,,notpresent,notpresent,114.0,87.0,5.2,139.0,3.7,12.1,,,,yes,no,no,poor,no,no,ck
18,60.0,100.0,1.025,0.0,3.0,,normal,notpresent,notpresent,263.0,27.0,1.3,135.0,4.3,12.7,37,11400
19,62.0,60.0,1.015,1.0,0.0,,abnormal,present,notpresent,100.0,31.0,1.6,,,10.3,30,5300,3.7,yes,no
20,61.0,80.0,1.015,2.0,0.0,abnormal,abnormal,notpresent,notpresent,173.0,148.0,3.9,135.0,5.2,7.7
21,60.0,90.0,,,,,notpresent,notpresent,,180.0,76.0,4.5,,10.9,32,6200,3.6,yes,yes,yes,good,no,no
22,48.0,80.0,1.025,4.0,0.0,normal,abnormal,notpresent,notpresent,95.0,163.0,7.7,136.0,3.8,9.8,32
23,21.0,70.0,1.01,0.0,0.0,,normal,notpresent,notpresent,,,,,,,,,no,no,no,poor,no,yes,ckd
24,42.0,100.0,1.015,4.0,0.0,normal,abnormal,notpresent,present,,50.0,1.4,129.0,4.0,11.1,39,8300,
Ln 1, Col 1 Spaces: 4 UTF-8 LF Plain Text
Desktop
31°C Haze
10:39
01-04-2023


File Edit Selection View Go Run Terminal Help
data1.csv - ckd - Visual Studio Code
EXPLORER
ckdp.py
data1.csv
CKD
lib
ckd.DOCX
ckdp.py
data.csv
data1.csv
OUTLINE
TIMELINE
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,classi
0,48.0,80.0,1.02,1.0,0.0,,normal,notpresent,notpresent,121.0,36.0,1.2,,,15.4,44,7800,5.2,yes,yes
1,7.0,50.0,1.02,4.0,0.0,,normal,notpresent,notpresent,,18.0,0.8,,,11.3,38,6000,,no,no,no,good,no
2,62.0,80.0,1.01,2.0,3.0,normal,normal,notpresent,notpresent,423.0,53.0,1.8,,,9.6,31,7500,,no,ye
3,48.0,70.0,1.005,4.0,0.0,normal,abnormal,present,notpresent,117.0,56.0,3.8,111.0,2.5,11.2,32,67
4,51.0,80.0,1.01,2.0,0.0,normal,normal,notpresent,notpresent,106.0,26.0,1.4,,,11.6,35,7300,4.6,n
5,60.0,90.0,1.015,3.0,0.0,,,notpresent,notpresent,74.0,25.0,1.1,142.0,3.2,12.2,39,7800,4.4,yes,y
6,68.0,70.0,1.01,0.0,0.0,,normal,notpresent,notpresent,100.0,54.0,24.0,104.0,4.0,12.4,36,,,no,no
7,24.0,,1.015,2.0,4.0,normal,abnormal,notpresent,notpresent,410.0,31.0,1.1,,,12.4,44,6900,5,no,y
8,52.0,100.0,1.015,3.0,0.0,normal,abnormal,present,notpresent,138.0,60.0,1.9,,,10.8,33,9600,4.0,
9,53.0,90.0,1.02,2.0,0.0,abnormal,abnormal,present,notpresent,70.0,107.0,7.2,114.0,3.7,9.5,29,12
10,50.0,60.0,1.01,2.0,4.0,,abnormal,present,notpresent,490.0,55.0,4.0,,,9.4,28,,,yes,yes,no,good
11,63.0,70.0,1.01,3.0,0.0,abnormal,abnormal,present,notpresent,380.0,60.0,2.7,131.0,4.2,10.8,32,
12,68.0,70.0,1.015,3.0,1.0,,normal,present,notpresent,208.0,72.0,2.1,138.0,5.8,9.7,28,12200,3.4,
13,68.0,70.0,,,,,notpresent,notpresent,98.0,86.0,4.6,135.0,3.4,9.8,,,,yes,yes,yes,poor,yes,no,c
14,68.0,80.0,1.01,3.0,2.0,normal,abnormal,present,present,157.0,90.0,4.1,130.0,6.4,5.6,16,11000,
15,40.0,80.0,1.015,3.0,0.0,,normal,notpresent,notpresent,76.0,162.0,9.6,141.0,4.9,7.6,24,3800,2.
16,47.0,70.0,1.015,2.0,0.0,,normal,notpresent,notpresent,99.0,46.0,2.2,138.0,4.1,12.6,,,,no,no,n
17,47.0,80.0,,,,,notpresent,notpresent,114.0,87.0,5.2,139.0,3.7,12.1,,,,yes,no,no,poor,no,no,ck
18,60.0,100.0,1.025,0.0,3.0,,normal,notpresent,notpresent,263.0,27.0,1.3,135.0,4.3,12.7,37,11400
19,62.0,60.0,1.015,1.0,0.0,,abnormal,present,notpresent,100.0,31.0,1.6,,,10.3,30,5300,3.7,yes,no
20,61.0,80.0,1.015,2.0,0.0,abnormal,abnormal,notpresent,notpresent,173.0,148.0,3.9,135.0,5.2,7.7
21,60.0,90.0,,,,,notpresent,notpresent,,180.0,76.0,4.5,,10.9,32,6200,3.6,yes,yes,yes,good,no,no
22,48.0,80.0,1.025,4.0,0.0,normal,abnormal,notpresent,notpresent,95.0,163.0,7.7,136.0,3.8,9.8,32
23,21.0,70.0,1.01,0.0,0.0,,normal,notpresent,notpresent,,,,,,,,,no,no,no,poor,no,yes,ckd
24,42.0,100.0,1.015,4.0,0.0,normal,abnormal,notpresent,present,,50.0,1.4,129.0,4.0,11.1,39,8300,
Ln 1, Col 1 Spaces: 4 UTF-8 LF Plain Text
Desktop
31°C Haze
10:39
01-04-2023

C. SAMPLE CODING

```
import pandas as pd
import numpy as np
import logging
import sklearn
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.impute import KNNImputer
from sklearn import preprocessing
from sklearn.preprocessing import StandardScaler


pd.pandas.set_option('display.max_columns', None)


dataset = pd.read_csv("data.csv")

print(dataset.head())

dataset = dataset.drop('id', axis=1)

print(dataset.shape)

print(dataset.isnull().sum())
```

```
print(dataset.describe())

print(dataset.dtypes)


print(dataset['rbc'].value_counts())

dataset['rbc'] = dataset['rbc'].replace(to_replace = {'normal' : 0, 'abnormal'
: 1})


print(dataset['pc'].value_counts())
dataset['pc'] = dataset['pc'].replace(to_replace = {'normal' : 0, 'abnormal' :
1})


print(dataset['pcc'].value_counts())
dataset['pcc']=dataset['pcc'].replace(to_replace=
{'notpresent':0,'present':1})


print(dataset['ba'].value_counts())
dataset['ba']=dataset['ba'].replace(to_replace= {'notpresent':0,'present':1})


print(dataset['htn'].value_counts())
dataset['htn'] = dataset['htn'].replace(to_replace = {'yes' : 1, 'no' : 0})
```

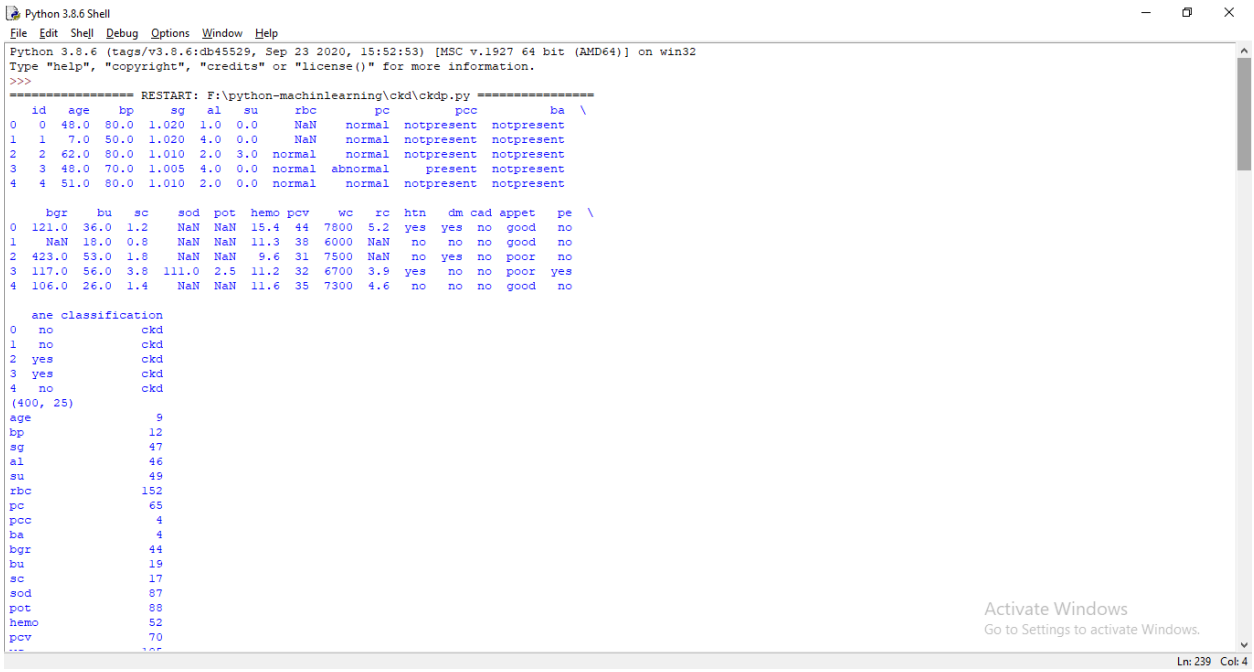
```
print(dataset['dm'].value_counts())
dataset['dm'] = dataset['dm'].replace(to_replace = {'\tyes':'yes', ' yes':'yes',
'\tno':'no'})
dataset['dm'] = dataset['dm'].replace(to_replace = {'yes' : 1, 'no' : 0})


print(dataset['cad'].value_counts())
dataset['cad'] = dataset['cad'].replace(to_replace = {'\tno':'no'})
dataset['cad'] = dataset['cad'].replace(to_replace = {'yes' : 1, 'no' : 0})


print(dataset['appet'].unique())
dataset['appet']=
dataset['appet'].replace(to_replace={'good':1,'poor':0,'no':np.nan})


print(dataset['pe'].value_counts())
dataset['pe'] = dataset['pe'].replace(to_replace = {'yes' : 1, 'no' : 0})



print(dataset['ane'].value_counts())
dataset['ane'] = dataset['ane'].replace(to_replace = {'yes' : 1, 'no' :0})


print(dataset['classification'].value_counts())
```

```
dataset['classification']=
dataset['classification'].replace(to_replace={'ckd\t':'ckd'})
dataset["classification"] = [1 if i == "ckd" else 0 for i in
dataset["classification"]]



print(dataset.head())

print(dataset.dtypes)

dataset['pcv'] = pd.to_numeric(dataset['pcv'], errors='coerce')
dataset['wc'] = pd.to_numeric(dataset['wc'], errors='coerce')
dataset['rc'] = pd.to_numeric(dataset['rc'], errors='coerce')
 print(dataset.dtypes)

#checking for missing values:

print(dataset.isnull().sum().sort_values(ascending=False))

print(dataset.columns)



features = ['age', 'bp', 'sg', 'al', 'su', 'rbc', 'pc', 'pcc', 'ba', 'bgr', 'bu',
          'sc', 'sod', 'pot', 'hemo', 'pcv', 'wc', 'rc', 'htn', 'dm', 'cad',
          'appet', 'pe', 'ane']
```

```
for feature in features:
    dataset[feature] = dataset[feature].fillna(dataset[feature].median())

print(dataset.isnull().any().sum())

plt.figure(figsize=(24,14))
sns.heatmap(dataset.corr(), annot=True, cmap='YlGnBu')
plt.show()


dataset.drop('pcv', axis=1, inplace=True)

dataset.head()
sns.countplot(dataset['classification'])

#Independent and Dependent feature:



from sklearn.linear_model import LogisticRegression


logreg = LogisticRegression()

X = dataset.iloc[:,:-1]
y = dataset['classification']
```

```
from sklearn.ensemble import ExtraTreesClassifier
import matplotlib.pyplot as plt
model=ExtraTreesClassifier()
model.fit(X,y)
plt.figure(figsize=(8,6))
ranked_features=pd.Series(model.feature_importances_,index=X.column
s)
ranked_features.nlargest(24).plot(kind='barh')
plt.show()

ranked_features.nlargest(8).index

X = dataset[['sg', 'htn', 'hemo', 'dm', 'al', 'appet', 'rc', 'pc']]
X.head()

X_train, X_test, y_train, y_test = train_test_split(X,y, stratify = y, shuffle
= True)
logreg.fit(X_train,y_train)

test_pred = logreg.predict(X_test)
train_pred = logreg.predict(X_train)
print(test_pred)

from sklearn.metrics import accuracy_score, confusion_matrix
print('Train Accuracy: ', accuracy_score(y_train, train_pred))
print('Test Accuracy: ', accuracy_score(y_test, test_pred))
```

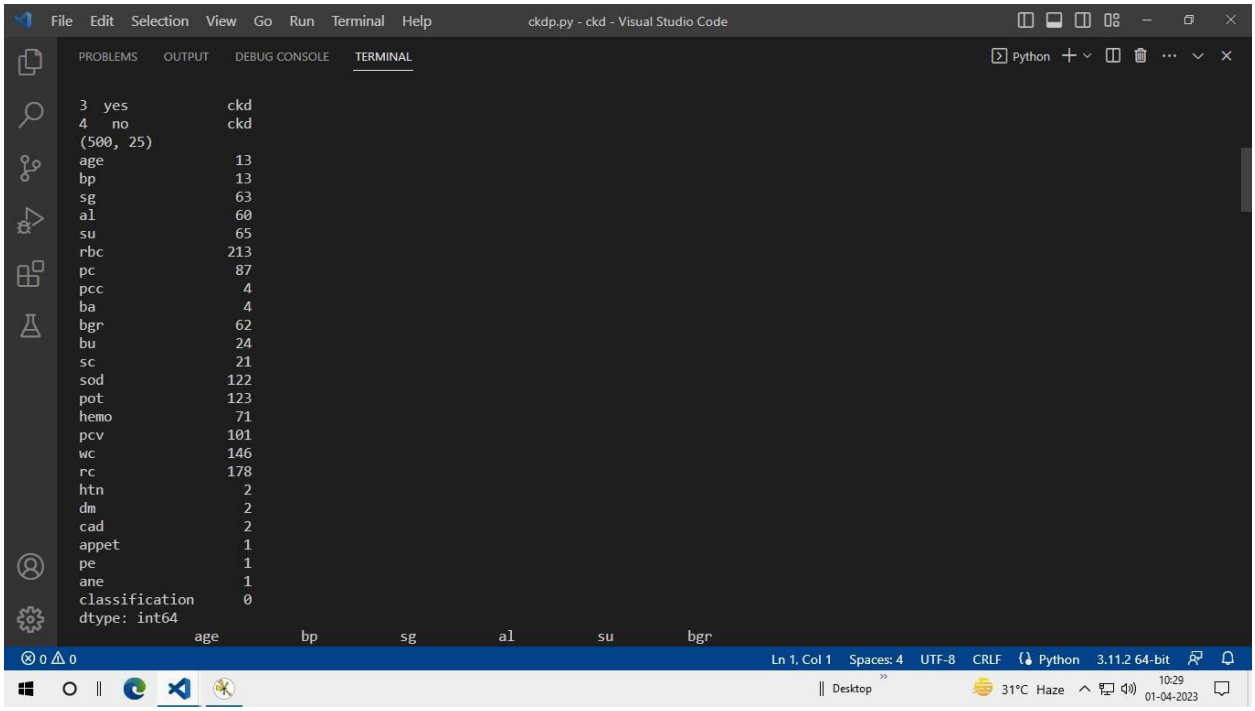
D. SAMPLE INPUT :

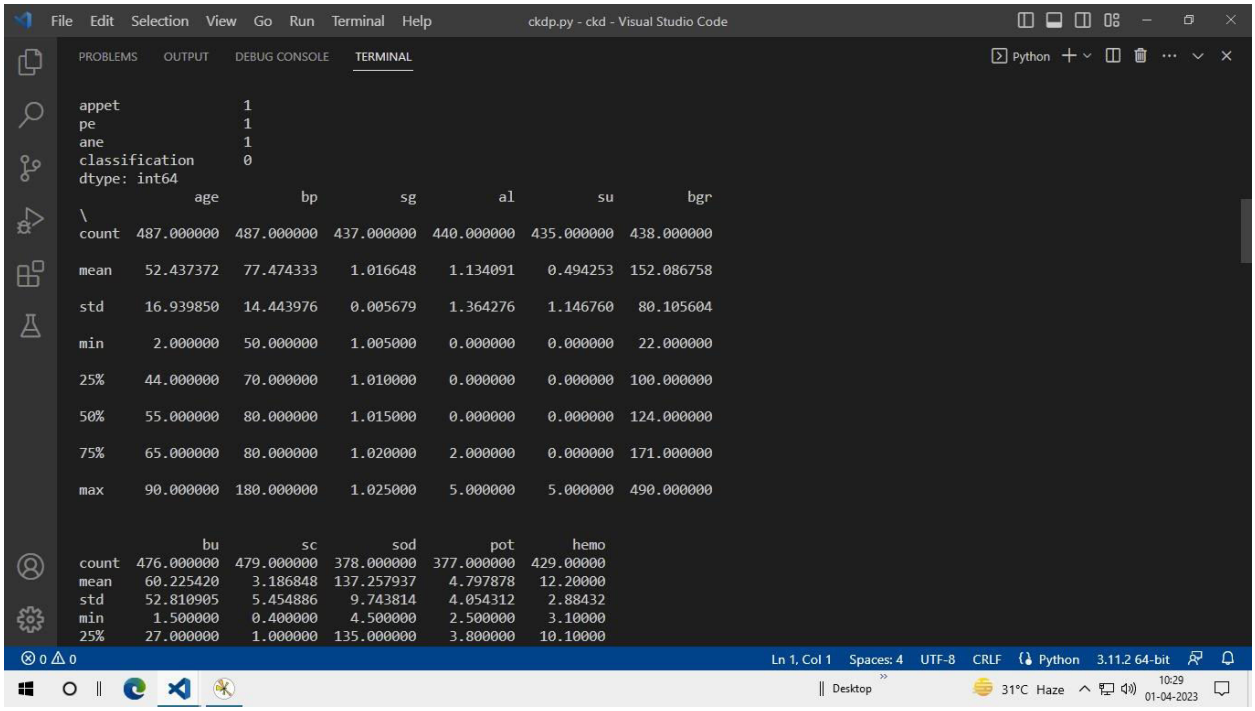


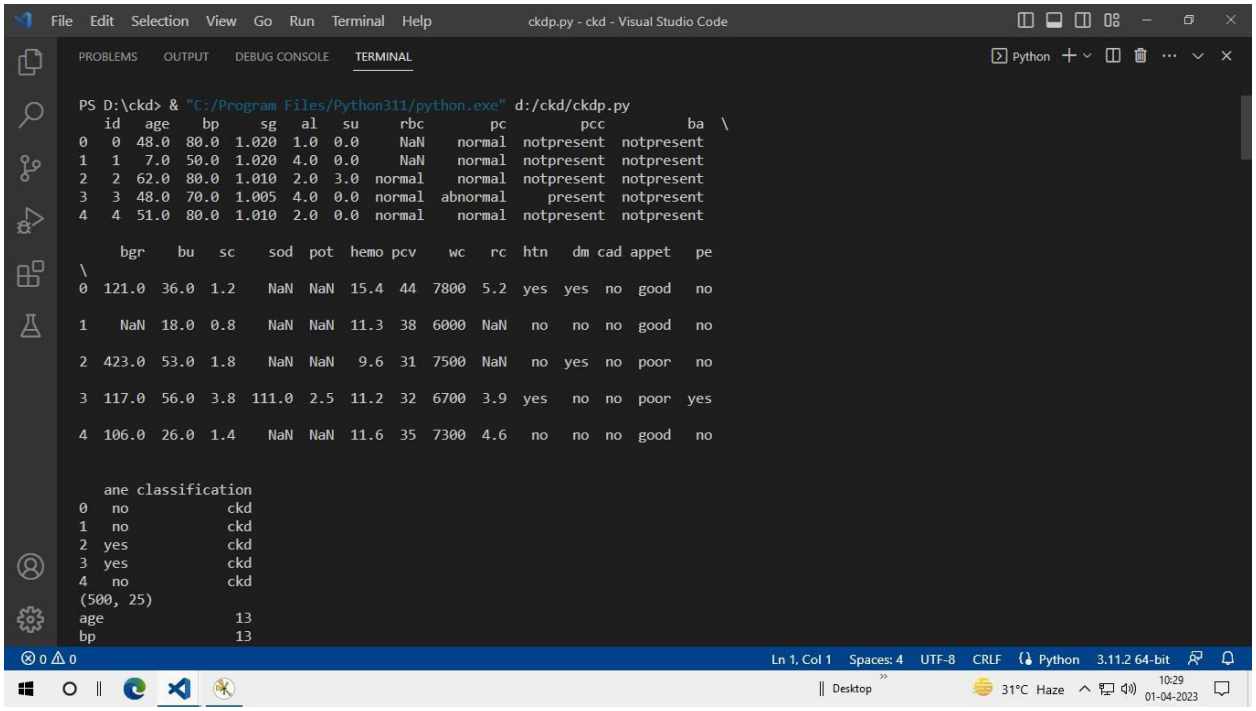
```
Python 3.8.6 Shell
File  Edit  Shell  Debug  Options  Window  Help
Python 3.8.6 (tags/v3.8.6:db45529, Sep 23 2020, 15:52:53) [MSC v.1927 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: F:\python-machinlearning\ckd\ckdp.py ==========
   id   age    bp     sg   al   su     rbc        pc         pcc          ba  \
0   0  48.0  80.0  1.020  1.0  0.0     NaN    normal  notpresent  notpresent
1   1   7.0  50.0  1.020  4.0  0.0     NaN    normal  notpresent  notpresent
2   2  62.0  80.0  1.010  2.0  3.0  normal    normal  notpresent  notpresent
3   3  48.0  70.0  1.005  4.0  0.0  normal  abnormal     present  notpresent
4   4  51.0  80.0  1.010  2.0  0.0  normal    normal  notpresent  notpresent

     bgr     bu   sc     sod  pot  hemo pcv    wc   rc  htn   dm cad appet   pe  \
0  121.0   36.0  1.2     NaN  NaN  15.4  44  7800  5.2  yes  yes  no  good   no
1    NaN   18.0  0.8     NaN  NaN  11.3  38  6000  NaN   no   no  no  good   no
2  423.0   53.0  1.8     NaN  NaN   9.6  31  7500  NaN   no  yes  no  poor   no
3  117.0   56.0  3.8   111.0  2.5  11.2  32  6700  3.9  yes   no  no  poor  yes
4  106.0   26.0  1.4     NaN  NaN  11.6  35  7300  4.6   no   no  no  good   no

   ane classification
0   no            ckd
1   no            ckd
2  yes            ckd
3  yes            ckd
4   no            ckd
(400, 25)
age                9
bp                12
sg                47
al                46
su                49
rbc              152
pc                65
pcc                4
ba                 4
bgr               44
bu                19
sc                17
sod               87
pot               88
hemo              52
pcv               70
                 Activate Windows
                 Go to Settings to activate Windows.
                                                                    Ln: 239  Col: 4
```

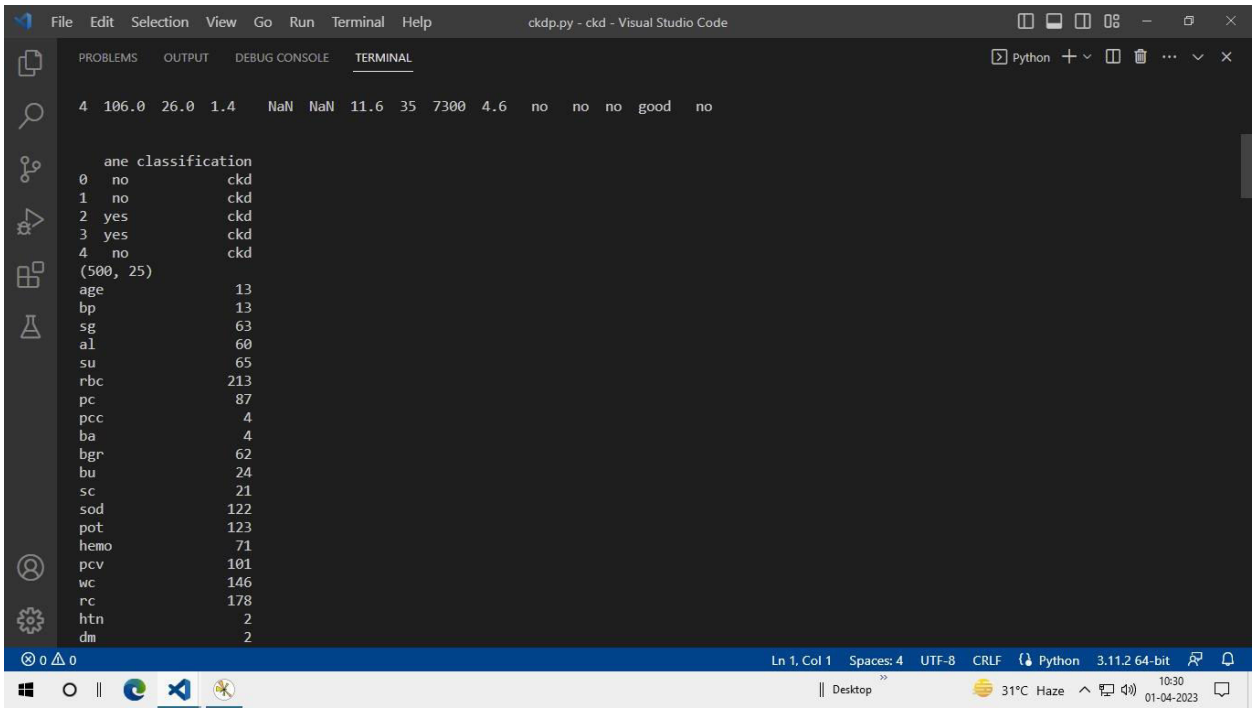
```
Python 3.8.6 Shell
File  Edit  Shell  Debug  Options  Window  Help
htn                2
dm                 2
cad                2
appet              1
pe                 1
ane                1
classification     0
dtype: int64
               age          bp          sg          al          su          bgr  \
count  391.000000  388.000000  353.000000  354.000000  351.000000  356.000000
mean    51.483376   76.469072    1.017408    1.016949    0.450142  148.036517
std     17.169714   13.683637    0.005717    1.352679    1.099191   79.281714
min      2.000000   50.000000    1.005000    0.000000    0.000000   22.000000
25%     42.000000   70.000000    1.010000    0.000000    0.000000   99.000000
50%     55.000000   80.000000    1.020000    0.000000    0.000000  121.000000
75%     64.500000   80.000000    1.020000    2.000000    0.000000  163.000000
max     90.000000  180.000000    1.025000    5.000000    5.000000  490.000000

               bu          sc         sod         pot        hemo
count  381.000000  383.000000  313.000000  312.000000  348.000000
mean    57.425722    3.072454  137.528754    4.627244   12.526437
std     50.503006    5.741126   10.408752    3.193904    2.912587
min      1.500000    0.400000    4.500000    2.500000    3.100000
25%     27.000000    0.900000  135.000000    3.800000   10.300000
50%     42.000000    1.300000  138.000000    4.400000   12.650000
75%     66.000000    2.800000  142.000000    4.900000   15.000000
max    391.000000   76.000000  163.000000   47.000000   17.800000
age              float64
bp               float64
sg               float64
al               float64
su               float64
rbc               object
pc                object
pcc               object
ba                object
bgr              float64
bu               float64
sc               float64
sod              float64
pot              float64
                 Activate Windows
                 Go to Settings to activate Windows.
                                                                    Ln: 239  Col: 4
```

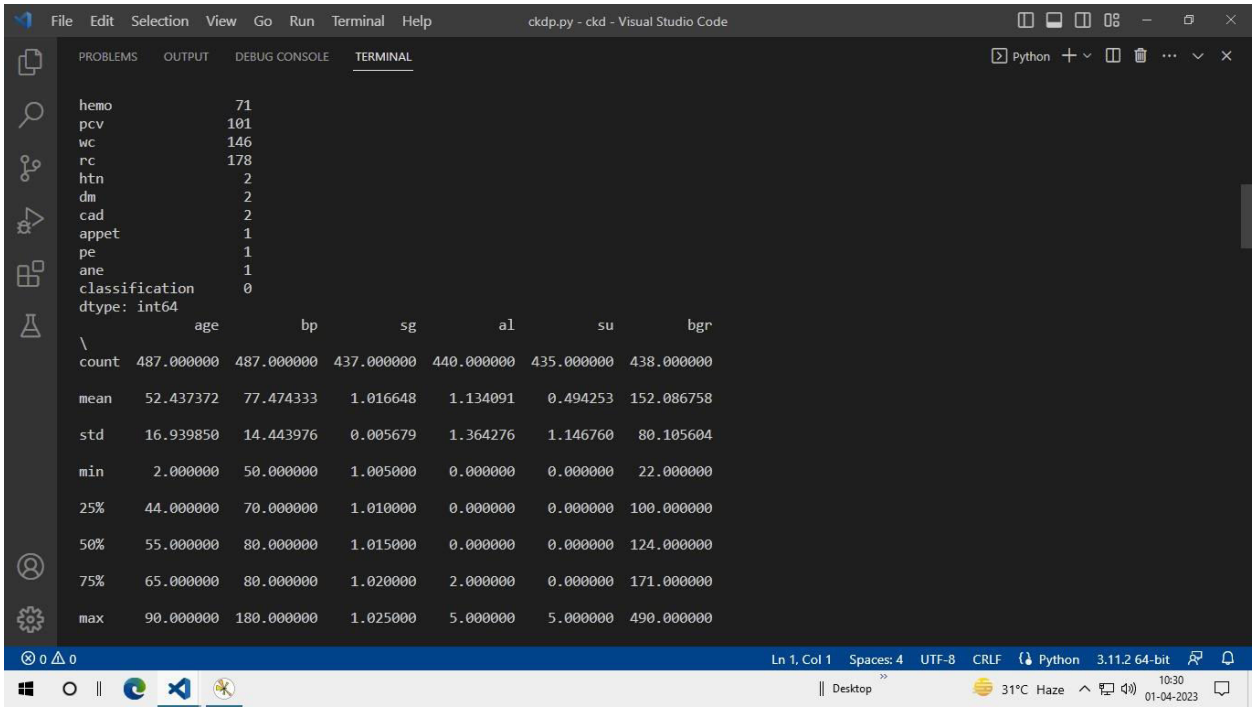
E. SAMPLE OUTPUT :

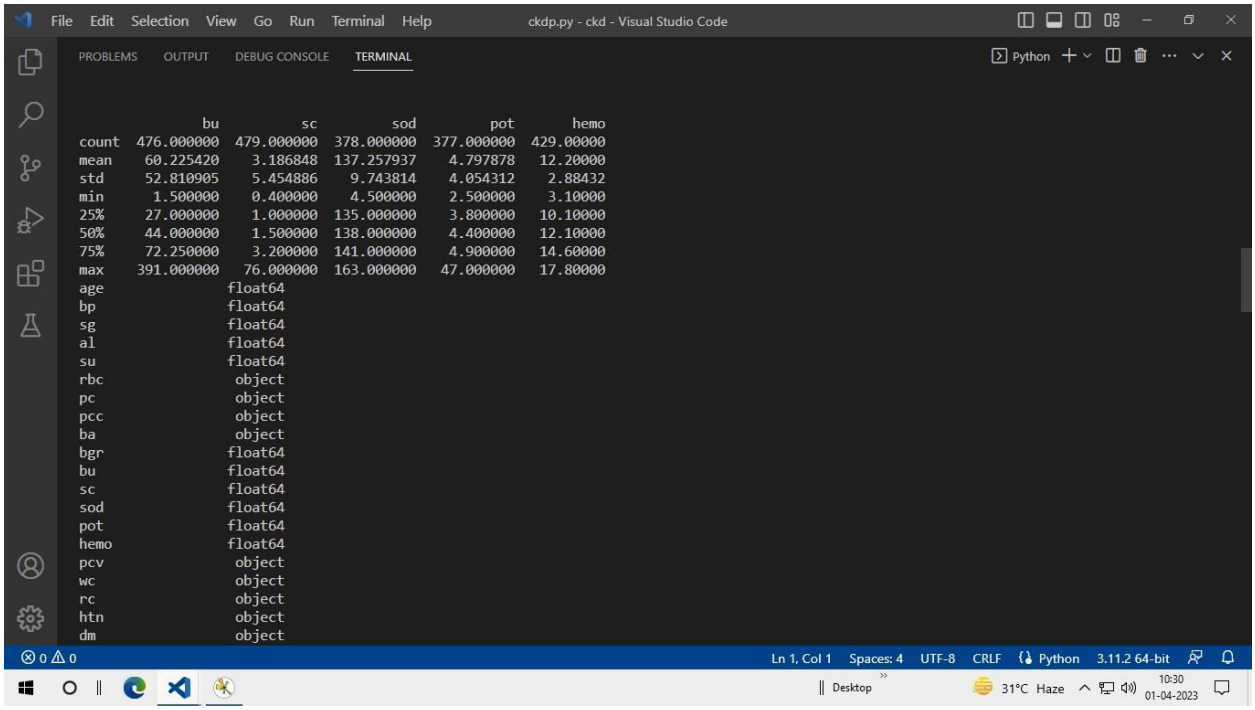


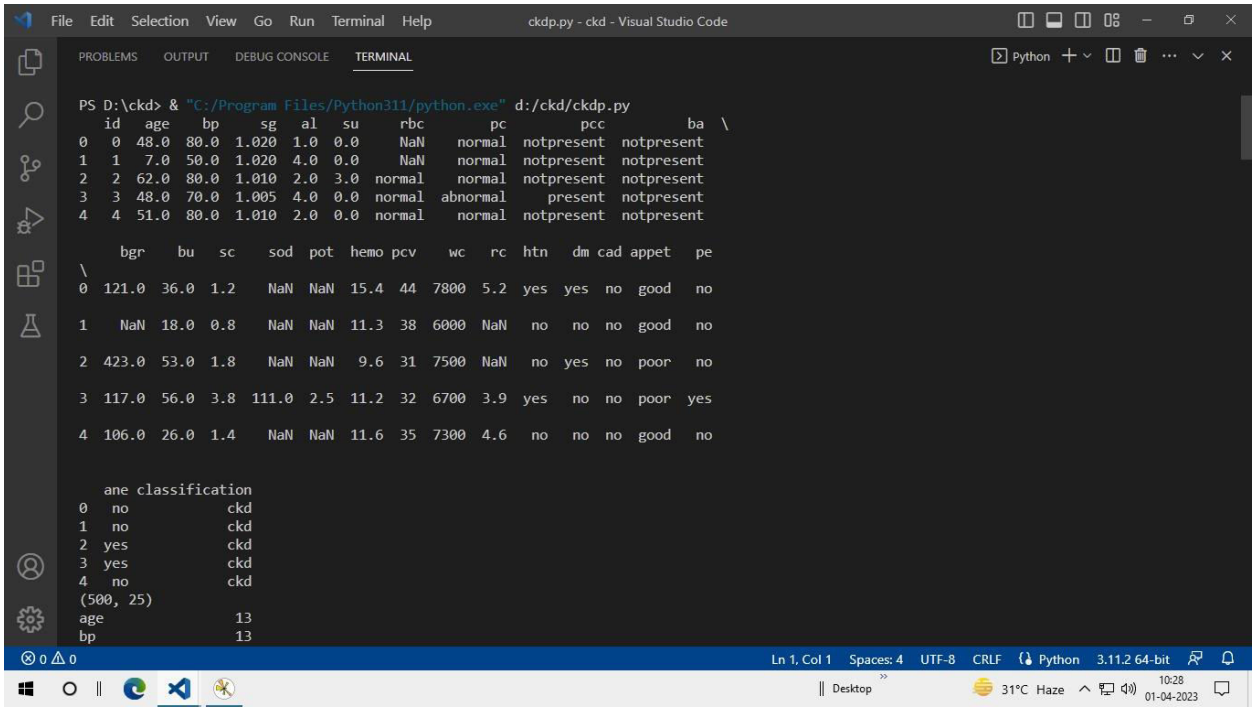


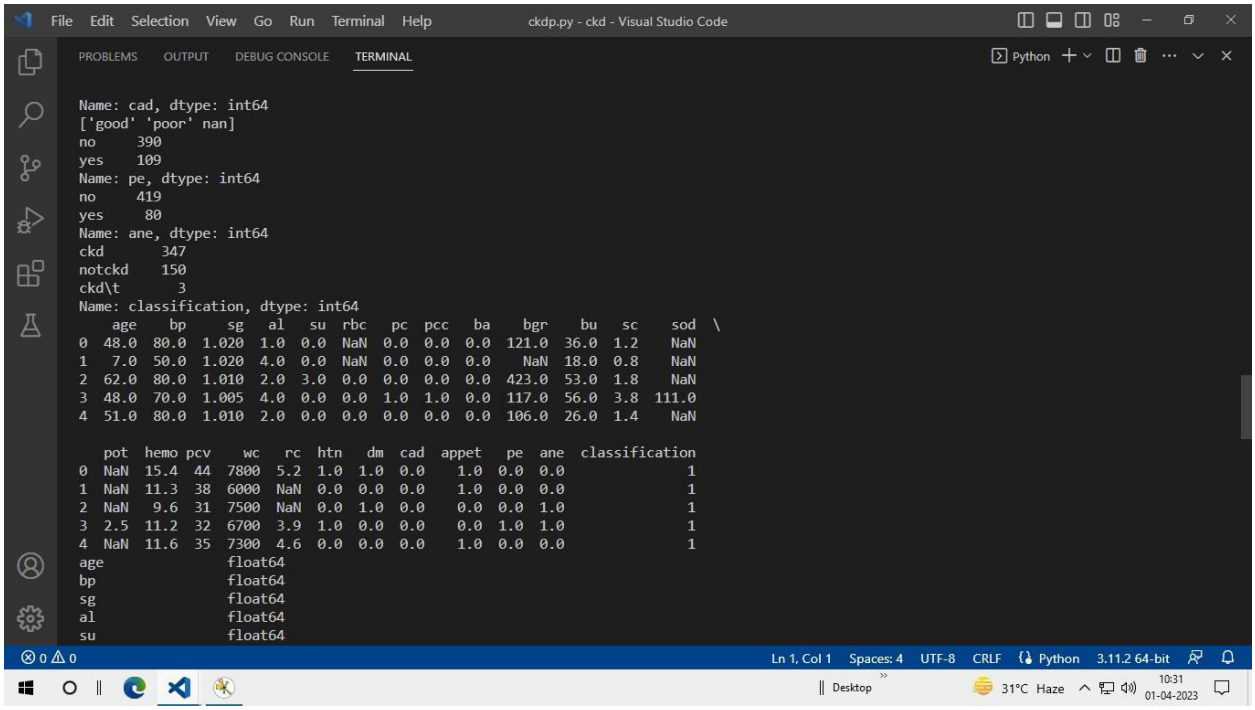

File Edit Selection View Go Run Terminal Help
ckdp.py - ckd - Visual Studio Code
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Python
PS D:\ckd> & "C:/Program Files/Python311/python.exe" d:/ckd/ckdp.py
id age bp sg al su rbc pc pcc ba \
0 0 48.0 80.0 1.020 1.0 0.0 NaN normal notpresent notpresent
1 1 7.0 50.0 1.020 4.0 0.0 NaN normal notpresent notpresent
2 2 62.0 80.0 1.010 2.0 3.0 normal normal notpresent notpresent
3 3 48.0 70.0 1.005 4.0 0.0 normal abnormal present notpresent
4 4 51.0 80.0 1.010 2.0 0.0 normal normal notpresent notpresent
bgr bu sc sod pot hemo pcv wc rc htn dm cad appet pe
\
0 121.0 36.0 1.2 NaN NaN 15.4 44 7800 5.2 yes yes no good no
1 NaN 18.0 0.8 NaN NaN 11.3 38 6000 NaN no no no good no
2 423.0 53.0 1.8 NaN NaN 9.6 31 7500 NaN no yes no poor no
3 117.0 56.0 3.8 111.0 2.5 11.2 32 6700 3.9 yes no no poor yes
4 106.0 26.0 1.4 NaN NaN 11.6 35 7300 4.6 no no no good no
ane classification
0 no ckd
1 no ckd
2 yes ckd
3 yes ckd
4 no ckd
(500, 25)
age 13
bp 13
Ln 1, Col 1 Spaces: 4 UTF-8 CRLF Python 3.11.2 64-bit
Desktop
31°C Haze
10:29
01-04-2023

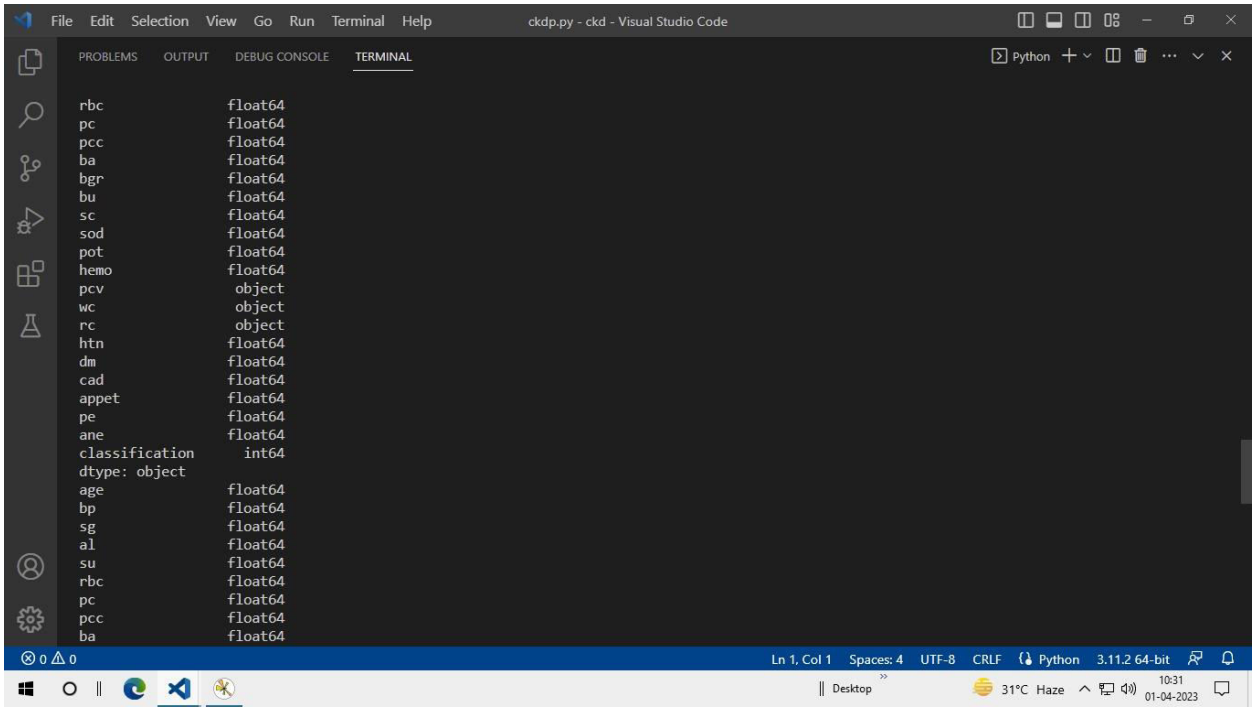

File Edit Selection View Go Run Terminal Help
ckdp.py - ckd - Visual Studio Code
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Python
4 106.0 26.0 1.4 NaN NaN 11.6 35 7300 4.6 no no no good no
ane classification
0 no ckd
1 no ckd
2 yes ckd
3 yes ckd
4 no ckd
(500, 25)
age 13
bp 13
sg 63
al 60
su 65
rbc 213
pc 87
pcc 4
ba 4
bgr 62
bu 24
sc 21
sod 122
pot 123
hemo 71
pcv 101
wc 146
rc 178
htn 2
dm 2
Ln 1, Col 1 Spaces: 4 UTF-8 CRLF Python 3.11.2 64-bit
Desktop
31°C Haze
10:30
01-04-2023

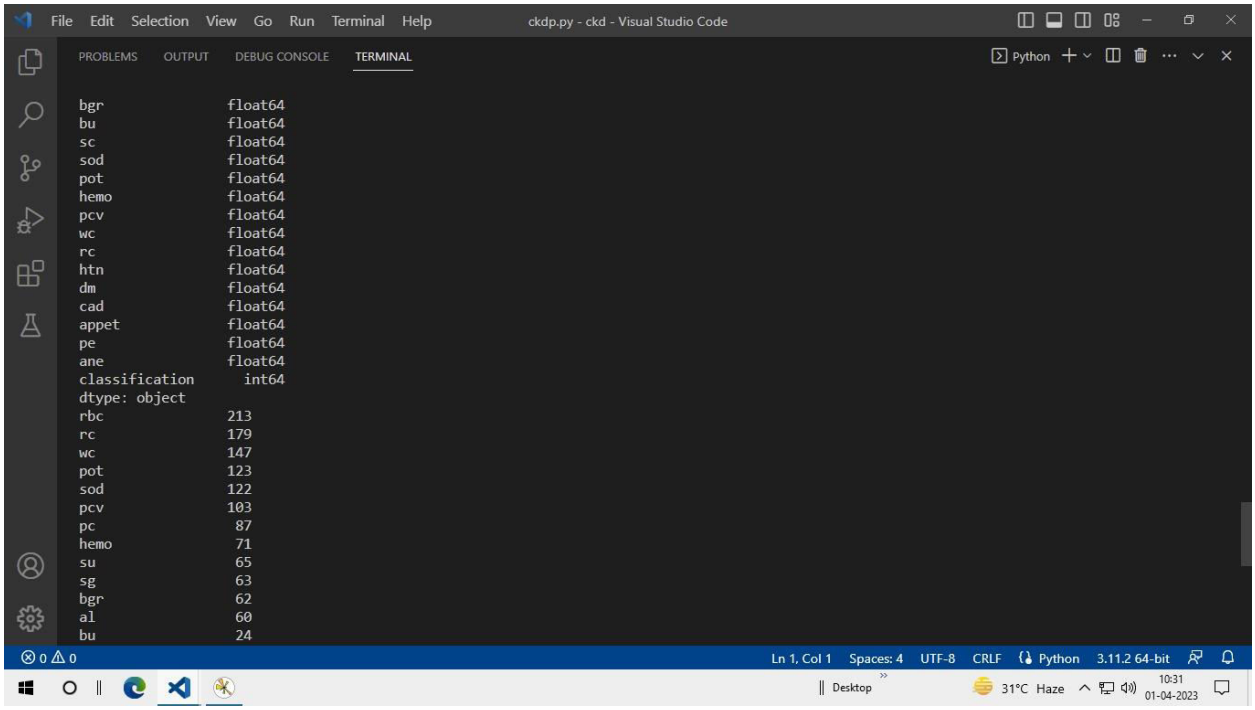

File Edit Selection View Go Run Terminal Help
ckdp.py - ckd - Visual Studio Code
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Python
hemo 71
pcv 101
wc 146
rc 178
htn 2
dm 2
cad 2
appet 1
pe 1
ane 1
classification 0
dtype: int64
age bp sg al su bgr
\
count 487.000000 487.000000 437.000000 440.000000 435.000000 438.000000
mean 52.437372 77.474333 1.016648 1.134091 0.494253 152.086758
std 16.939850 14.443976 0.005679 1.364276 1.146760 80.105604
min 2.000000 50.000000 1.005000 0.000000 0.000000 22.000000
25% 44.000000 70.000000 1.010000 0.000000 0.000000 100.000000
50% 55.000000 80.000000 1.015000 0.000000 0.000000 124.000000
75% 65.000000 80.000000 1.020000 2.000000 0.000000 171.000000
max 90.000000 180.000000 1.025000 5.000000 5.000000 490.000000
Ln 1, Col 1 Spaces: 4 UTF-8 CRLF Python 3.11.2 64-bit
Desktop
31°C Haze
10:30
01-04-2023

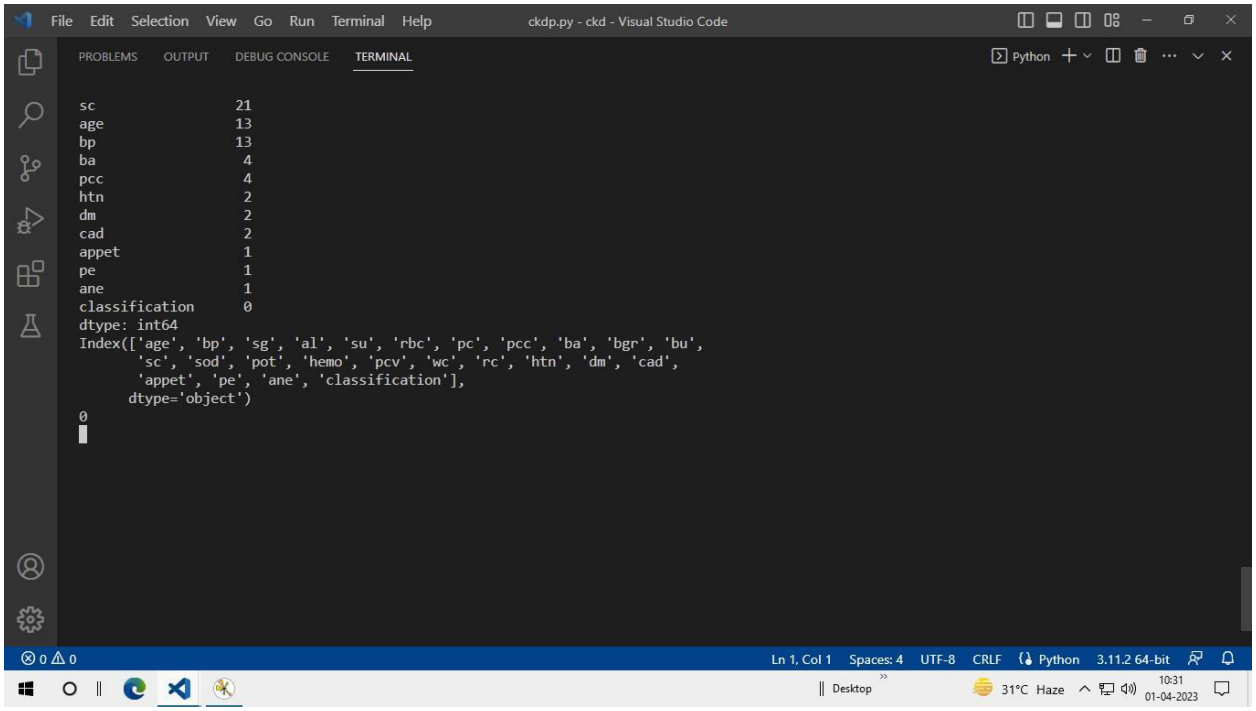

File Edit Selection View Go Run Terminal Help
ckdp.py - ckd - Visual Studio Code
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Python
bu sc sod pot hemo
count 476.000000 479.000000 378.000000 377.000000 429.00000
mean 60.225420 3.186848 137.257937 4.797878 12.20000
std 52.810905 5.454886 9.743814 4.054312 2.88432
min 1.500000 0.400000 4.500000 2.500000 3.10000
25% 27.000000 1.000000 135.000000 3.800000 10.10000
50% 44.000000 1.500000 138.000000 4.400000 12.10000
75% 72.250000 3.200000 141.000000 4.900000 14.60000
max 391.000000 76.000000 163.000000 47.000000 17.80000
age float64
bp float64
sg float64
al float64
su float64
rbc object
pc object
pcc object
ba object
bgr float64
bu float64
sc float64
sod float64
pot float64
hemo float64
pcv object
wc object
rc object
htn object
dm object
Ln 1, Col 1 Spaces: 4 UTF-8 CRLF Python 3.11.2 64-bit
Desktop
31°C Haze
10:30
01-04-2023


File Edit Selection View Go Run Terminal Help
ckdp.py - ckd - Visual Studio Code
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Python
PS D:\ckd> & "C:/Program Files/Python311/python.exe" d:/ckd/ckdp.py
id age bp sg al su rbc pc pcc ba \
0 0 48.0 80.0 1.020 1.0 0.0 NaN normal notpresent notpresent
1 1 7.0 50.0 1.020 4.0 0.0 NaN normal notpresent notpresent
2 2 62.0 80.0 1.010 2.0 3.0 normal normal notpresent notpresent
3 3 48.0 70.0 1.005 4.0 0.0 normal abnormal present notpresent
4 4 51.0 80.0 1.010 2.0 0.0 normal normal notpresent notpresent
bgr bu sc sod pot hemo pcv wc rc htn dm cad appet pe
\
0 121.0 36.0 1.2 NaN NaN 15.4 44 7800 5.2 yes yes no good no
1 NaN 18.0 0.8 NaN NaN 11.3 38 6000 NaN no no no good no
2 423.0 53.0 1.8 NaN NaN 9.6 31 7500 NaN no yes no poor no
3 117.0 56.0 3.8 111.0 2.5 11.2 32 6700 3.9 yes no no poor yes
4 106.0 26.0 1.4 NaN NaN 11.6 35 7300 4.6 no no no good no
ane classification
0 no ckd
1 no ckd
2 yes ckd
3 yes ckd
4 no ckd
(500, 25)
age 13
bp 13
Ln 1, Col 1 Spaces: 4 UTF-8 CRLF Python 3.11.2 64-bit
Desktop
31°C Haze
10:28
01-04-2023


File Edit Selection View Go Run Terminal Help
ckdp.py - ckd - Visual Studio Code
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Python
Name: cad, dtype: int64
['good' 'poor' nan]
no 390
yes 109
Name: pe, dtype: int64
no 419
yes 80
Name: ane, dtype: int64
ckd 347
notckd 150
ckd\t 3
Name: classification, dtype: int64
age bp sg al su rbc pc pcc ba bgr bu sc sod \
0 48.0 80.0 1.020 1.0 0.0 NaN 0.0 0.0 0.0 121.0 36.0 1.2 NaN
1 7.0 50.0 1.020 4.0 0.0 NaN 0.0 0.0 0.0 NaN 18.0 0.8 NaN
2 62.0 80.0 1.010 2.0 3.0 0.0 0.0 0.0 0.0 423.0 53.0 1.8 NaN
3 48.0 70.0 1.005 4.0 0.0 0.0 1.0 1.0 0.0 117.0 56.0 3.8 111.0
4 51.0 80.0 1.010 2.0 0.0 0.0 0.0 0.0 0.0 106.0 26.0 1.4 NaN
pot hemo pcv wc rc htn dm cad appet pe ane classification
0 NaN 15.4 44 7800 5.2 1.0 1.0 0.0 1.0 0.0 0.0 1
1 NaN 11.3 38 6000 NaN 0.0 0.0 0.0 1.0 0.0 0.0 1
2 NaN 9.6 31 7500 NaN 0.0 1.0 0.0 0.0 0.0 1.0 1
3 2.5 11.2 32 6700 3.9 1.0 0.0 0.0 0.0 1.0 1.0 1
4 NaN 11.6 35 7300 4.6 0.0 0.0 0.0 1.0 0.0 0.0 1
age float64
bp float64
sg float64
al float64
su float64
Ln 1, Col 1 Spaces: 4 UTF-8 CRLF Python 3.11.2 64-bit
Desktop
31°C Haze
10:31
01-04-2023


File Edit Selection View Go Run Terminal Help
ckdp.py - ckd - Visual Studio Code
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Python
rbc float64
pc float64
pcc float64
ba float64
bgr float64
bu float64
sc float64
sod float64
pot float64
hemo float64
pcv object
wc object
rc object
htn float64
dm float64
cad float64
appet float64
pe float64
ane float64
classification int64
dtype: object
age float64
bp float64
sg float64
al float64
su float64
rbc float64
pc float64
pcc float64
ba float64
Ln 1, Col 1 Spaces: 4 UTF-8 CRLF Python 3.11.2 64-bit
Desktop
31°C Haze
10:31
01-04-2023


File Edit Selection View Go Run Terminal Help
ckdp.py - ckd - Visual Studio Code
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Python
bgr float64
bu float64
sc float64
sod float64
pot float64
hemo float64
pcv float64
wc float64
rc float64
htn float64
dm float64
cad float64
appet float64
pe float64
ane float64
classification int64
dtype: object
rbc 213
rc 179
wc 147
pot 123
sod 122
pcv 103
pc 87
hemo 71
su 65
sg 63
bgr 62
al 60
bu 24
Ln 1, Col 1 Spaces: 4 UTF-8 CRLF Python 3.11.2 64-bit
Desktop
31°C Haze
10:31
01-04-2023


File Edit Selection View Go Run Terminal Help
ckdp.py - ckd - Visual Studio Code
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Python
sc 21
age 13
bp 13
ba 4
pcc 4
htn 2
dm 2
cad 2
appet 1
pe 1
ane 1
classification 0
dtype: int64
Index(['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'],
dtype='object')
0
Ln 1, Col 1 Spaces: 4 UTF-8 CRLF Python 3.11.2 64-bit
Desktop
31°C Haze
10:31
01-04-2023

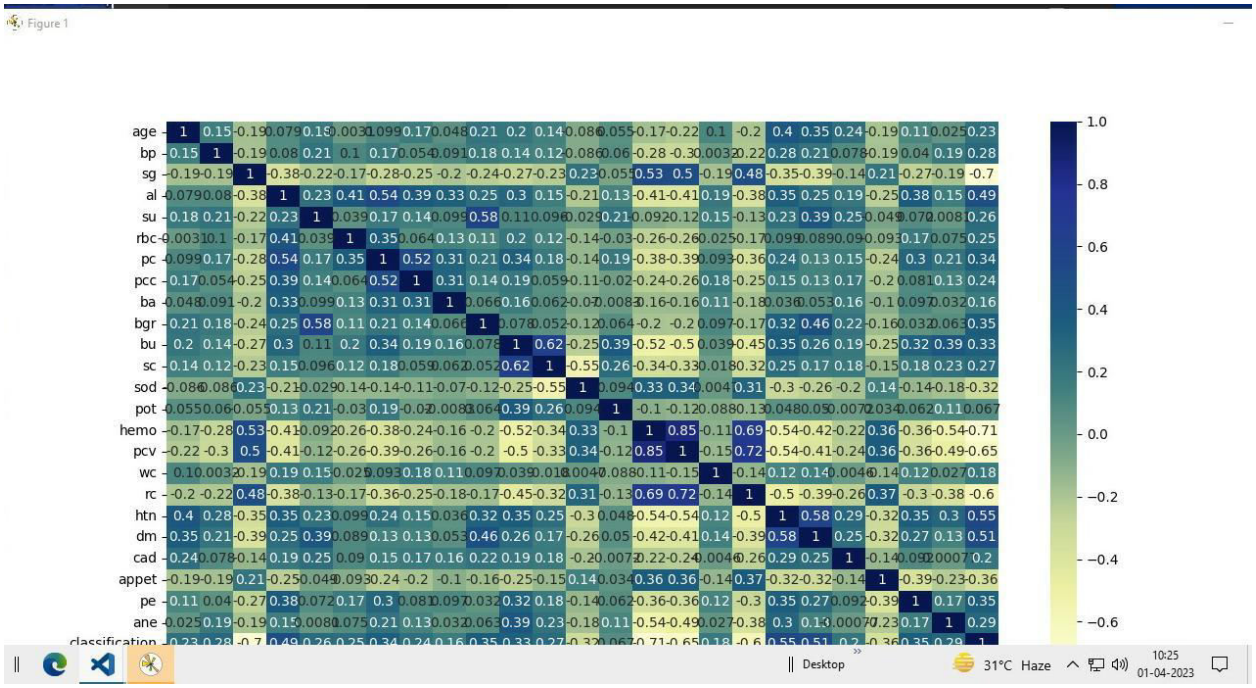


Figure 1
age
bp
sg
al
su
rbc
pc
pcc
ba
bgr
bu
sc
sod
pot
hemo
pcv
wc
rc
htn
dm
cad
appet
pe
ane
classification
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
Desktop
31°C Haze
10:25
01-04-2023