A Project Report

On

EARLY DETECTION OF CHRONIC KIDNEY DISEASE USING MACHINE LEARNING

Submitted in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

Under the Guidance of

Dr. S. SUJANTHI M.E., PH.D.,
Assistant Professor/CSE

Submitted by

TEAM ID: PNT2022TMID15499

927619BCS4078 - MONISHSURYA S M
927619BCS4021 - DEEPAKRAJ K
927619BCS4085 - NAVEEN KUMAR P
927619BCS4098 - RANJITH KUMAR P

NAALAIYA THIRAN – EXPERIENTIAL PROJECT BASED LEARNING INITIATIVE

18CSE040L - PROFESSIONAL READINESS FOR INNOVATION, EMPLOYABILITY AND ENTERPRENURSHIP

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

M.KUMARASAMY COLLEGE OF ENGINEERING, KARUR
(Autonomous)
Karur - 639 113
November, 2022

TABLE OF CONTENTS

ABSTRACT

Chronic Kidney Disease is a serious lifelong condition that induced by either kidney pathology or reduced kidney functions. Early prediction and proper treatments can possibly stop or slow the progression of this chronic disease to end-stage, where dialysis or kidney transplantation is the only way to save patient's life. In our project, we examine the ability of several machine-learning methods for early prediction of chronic kidney disease. This matter has been studied widely; however, we are supporting our methodology using predictive analytics, in which we examine the relationship in between data parameters as well as with the target class attribute. Predictive analytics enables us to introduce the optimal subset of parameters to feed machine learning to build a set of predictive models.

CHAPTER 1

INTRODUCTION

1.1 Project Overview

Kidney diseases avert the normal function of the Kidney. Mainly due to the large amount of alcohol consumption kidney disease arises. Early prediction of kidney disease using classification algorithms is an efficacious task that can help the doctors to diagnose the disease within a short duration of time. Discovering the existence of kidney disease at an early stage is a complex task for the doctors. The main objective of this project is to analyze the parameters of various classification algorithms and compare their predictive accuracies to find out the best classifier for determining the kidney disease. This Project examines data from Kidney patients concentrating on relationships between a key list of Kidney enzymes, proteins, age and gender using them to try and predict the likeliness of kidney disease. Here we are building a model by applying various machine learning algorithms find the best accurate model. And integrate to flask-based web application. User can predict the disease by entering parameters in the web application.

1.2 PURPOSE

Current screening strategies for kidney disease focus on detection of subclinical advanced kidney fibrosis but cannot identify those at high future risk of severe kidney disease. Our aim was to develop and validate a risk prediction model for incident kidney disease in the general population based on widely available factors. Kidney disease often progresses silently without symptoms and thus the diagnosis is often delayed until severe complications occur and prognosis becomes poor. In order to identify individuals in the general population who have a high risk of developing severe liver disease in the future, we developed and validated a Liver Disease risk prediction score, based on age, sex, alcohol use, waist-hip ratio, diabetes, and smoking, with or without measurement of the liver enzyme gamma-glutamyltransferase. The Kidney Disease score can be used as part of health counselling, and for planning further kidney investigations and follow-up.

CHAPTER 2

LITERATURE SURVEY

2.1 Existing Problem

Author	Year	Title	Algorithm used	Limitations
Andressa C.M. da Silveira	2022	Exploring Early Prediction of Chronic Kidney Disease Using Machine Learning Algorithms	Decision tree (DT), random forest, and multi- class Ad Boosted DTs	Leads to processing limitations, mainly for the ensemble models
Rayan Alazani	2022	Identification and Prediction of Chronic Diseases Using Machine Learning Approach	Convolutional neural network (CNN), K-nearest neighbor (KNN)	Identify and predict the patients with more common chronic illnesses
Reshma S	2020	Chronic Kidney Disease Using Machine Learning	Chronic kidney, SVM, Ant colony optimization	Slow disease progression, reduce complications of decreased Glomerular Filtration Rate (GFR)
Tauja K J	2019	Detection of Chronic Kidney Disease Using Machine Learning Techniques	CKD, Decision Tree, SVM, Random Forest, Naive Bayes	The strength of the data is not higher because of the size of the dataset
Deepika Bidri	2018	Early Prediction of Chronic Kidney Disease by using Machine Learning Techniques	Naive bayes; K-Nearest neighbor ; Machine learning	Leads to low accuracy

Table 2.1 - Existing Problem

2.2 References

1. Andressa C.M. da Silveira, Exploring Early Prediction of Chronic Kidney Disease Using Machine Learning Algorithms, January 2022 .
2. Rayan Alazani, Identification and Prediction of Chronic Diseases Using Machine Learning Approach, February 2022 .
3. Reshma S, Chronic Kidney Disease Prediction using Machine Learning, July 2020 .
4. Tauja K J. Detection of Chronic Kidney Disease Using Machine Learning Techniques, March 2019 .
5. Deepika Badri. Early Prediction of Chronic Kidney Disease by using Machine Learning Techniques, September 2018.

2.3 Problem Statement Definition

Kidney diseases avert the normal function of the Kidney. Mainly due to the large amount of alcohol consumption kidney disease arises. Early prediction of kidney disease using classification algorithms is an efficacious task that can help the doctors to diagnose the disease with in a short duration of time. Discovering the existence of Kidney disease at an early stage is a complex task for the doctors. The main objective of this project is to analyze the parameters of various classification algorithms and compare their predictive accuracies so as to find out the best classifier for determining the kidney disease. This Project examines data from Kidney patients concentrating on relationships between a key list of Kidney enzymes, proteins, age and gender using them to try and predict the likeliness of kidney disease. Here we are building a model by applying various machine learning algorithms find the best accurate model. And integrate to flask based web application. User can predict the disease by entering parameters in the web application.

CHAPTER 3

IDEATION AND PROPOSED SOLUTION

Ideation is the process where you generate ideas and solutions through sessions such as Sketching, Prototyping, Brainstorming, Brainwriting, Worst Possible Idea, and a wealth of other ideation techniques. Ideation is also the third stage in the Design Thinking process. In this project the ideation phase consists of,

3.1 Empathy Map and Canvas

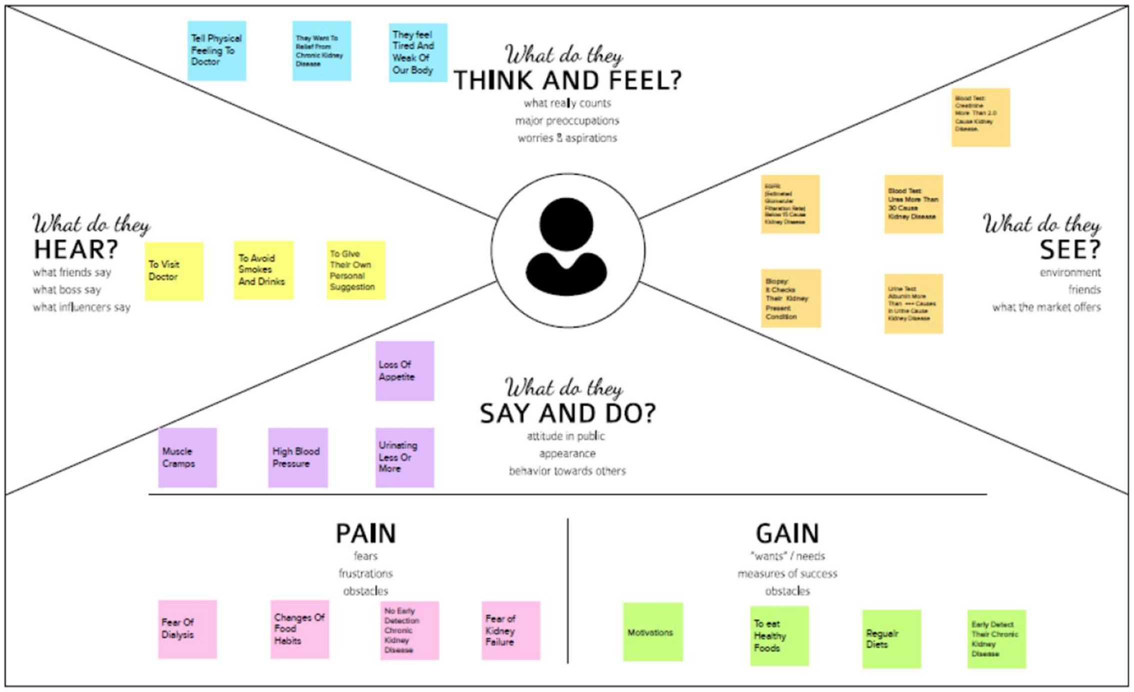
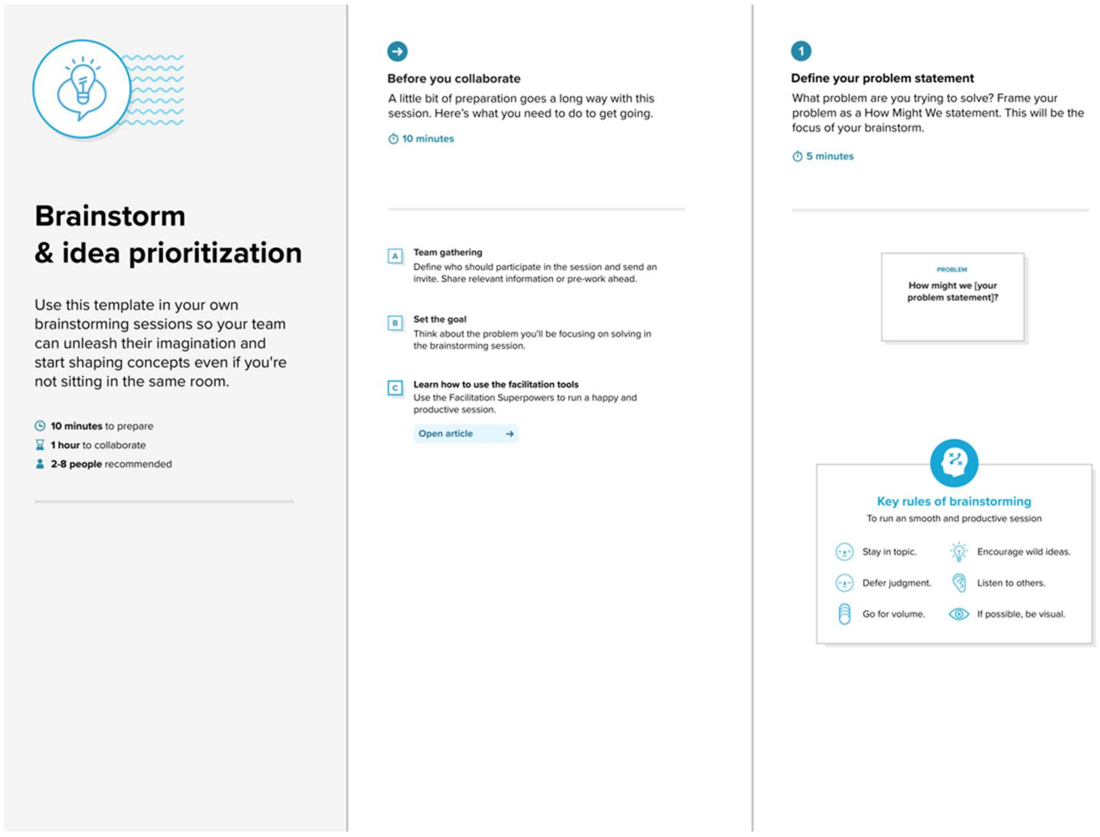

Tell Physical Feeling To Doctor
They Want To Relief From Chronic Kidney Disease
They feel Tired And Weak Of Our Body
What do they THINK AND FEEL?
what realy counts
major preoccupations
worries & aspirations
Blood Test: Creatinine More Than 2.0 Cause Kidney Disease.
What do they HEAR?
what friends say
what boss say
what influencers say
To Visit Doctor
To Avoid Smokes And Drinks
To Give Their Own Personal Suggestion
EGFR (Estimated Glomerular Filtration Rate) Below 15 Cause Kidney Disease
Blood Test: Urea More Than 30 Cause Kidney Disease
What do they SEE?
environment
friends
what the market offers
Biopsy: It Checks Their Kidney Present Condition
Urine Test: Albumin More Than +++ Causes In Urine Cause Kidney Disease
Loss Of Appetite
What do they SAY AND DO?
attitude in public
appearance
behavior towards others
Muscle Cramps
High Blood Pressure
Urinating Less Or More
PAIN
fears
frustrations
obstacles
Fear Of Dialysis
Changes Of Food Habits
No Early Detection Chronic Kidney Disease
Fear of Kidney Failure
GAIN
"wants" / needs
measures of success
obstacles
Motivations
To eat Healthy Foods
Regualr Diets
Early Detect Their Chronic Kidney Disease

Figure 3.1 - Empathy Map

3.2 Ideation and proposed solution

Step-1: Team Gathering, Collaboration and Select the Problem Statement



Brainstorm & idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

10 minutes to prepare
1 hour to collaborate
2-8 people recommended

Before you collaborate

A little bit of preparation goes a long way with this session. Here's what you need to do to get going.

10 minutes

A **Team gathering**
Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.

B **Set the goal**
Think about the problem you'll be focusing on solving in the brainstorming session.

C **Learn how to use the facilitation tools**
Use the Facilitation Superpowers to run a happy and productive session.

Open article →

1 **Define your problem statement**

What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.

5 minutes

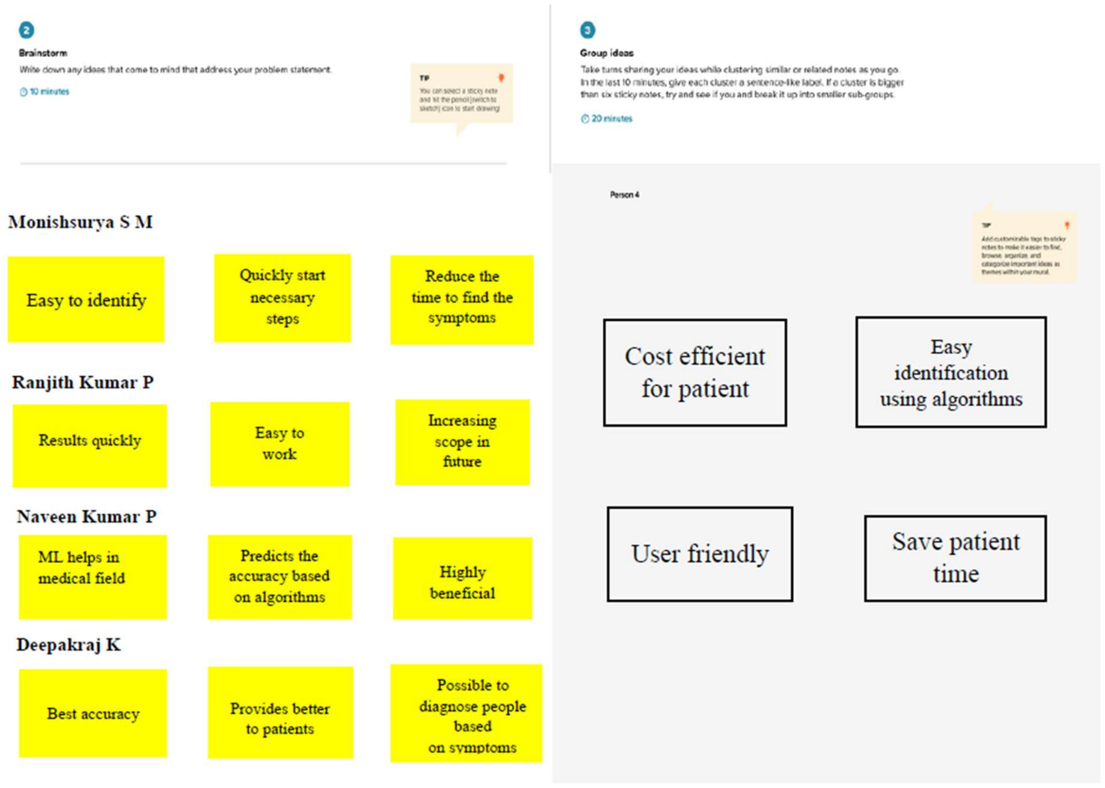
PROBLEM
How might we [your problem statement]?

Key rules of brainstorming
To run an smooth and productive session

Stay in topic.	Encourage wild ideas.
Defer judgment.	Listen to others.
Go for volume.	If possible, be visual.

Figure 3.2.1 - Brainstorm Techniques

Step-2: Brainstorm, Idea Listing and Grouping



2

Brainstorm

Write down any ideas that come to mind that address your problem statement.

10 minutes

TIP

You can select a sticky note and hit the pencil [switch to sketch] icon to start drawing!

Monishsurya S M

Easy to identify

Quickly start necessary steps

Reduce the time to find the symptoms

Ranjith Kumar P

Results quickly

Easy to work

Increasing scope in future

Naveen Kumar P

ML helps in medical field

Predicts the accuracy based on algorithms

Highly beneficial

Deepakraj K

Best accuracy

Provides better to patients

Possible to diagnose people based on symptoms

3

Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. In the last 10 minutes, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.

20 minutes

Person 4

TIP

Add customizable tags to sticky notes to make it easier to find, browse, organize, and categorize important ideas as themes within your mural.

Cost efficient for patient

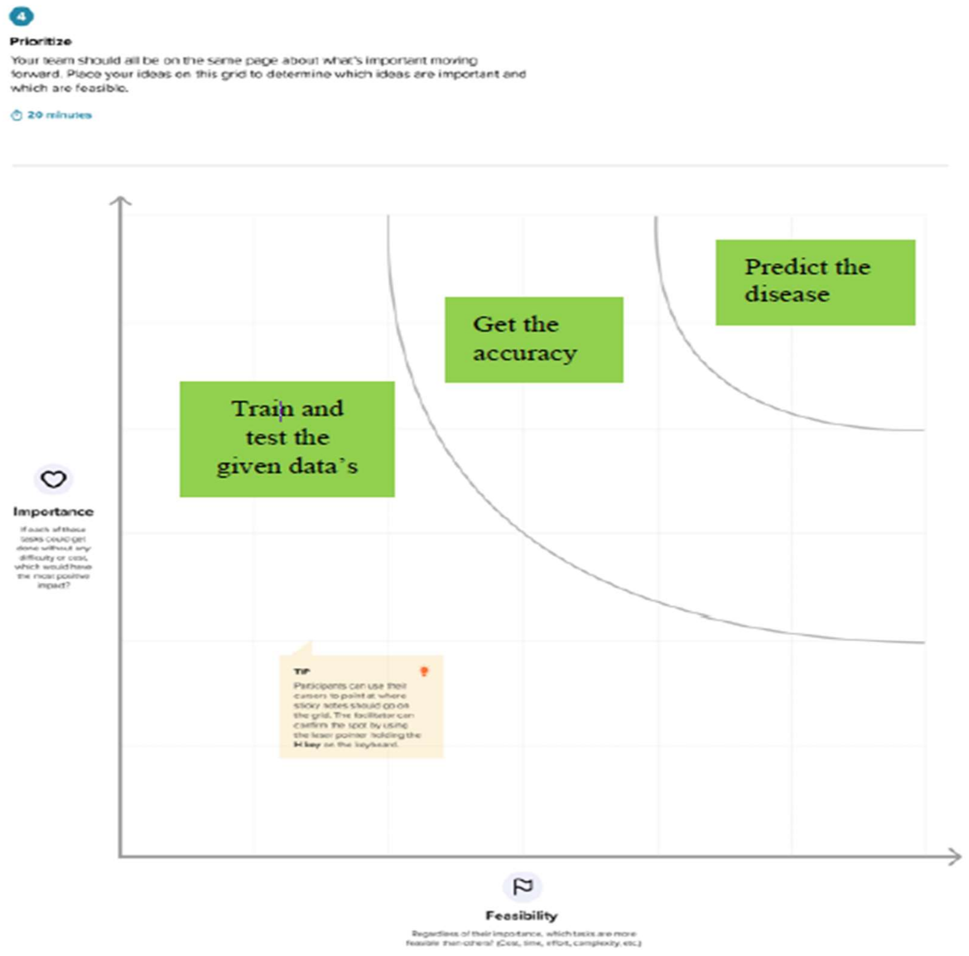
Easy identification using algorithms

User friendly

Save patient time

Figure 3.2.2 - Brainstorm, Idea Listing and Grouping

Step-3: Idea Prioritization



Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

If each of these tasks could get done without any difficulty or cost, which would have the most positive impact?

Participants can use their cursors to point at where sticky notes should go on the grid. The facilitator can confirm the spot by using the laser pointer holding the **H key** on the keyboard.

Regardless of their importance, which tasks are more feasible than others? (Cost, time, effort, complexity, etc.)

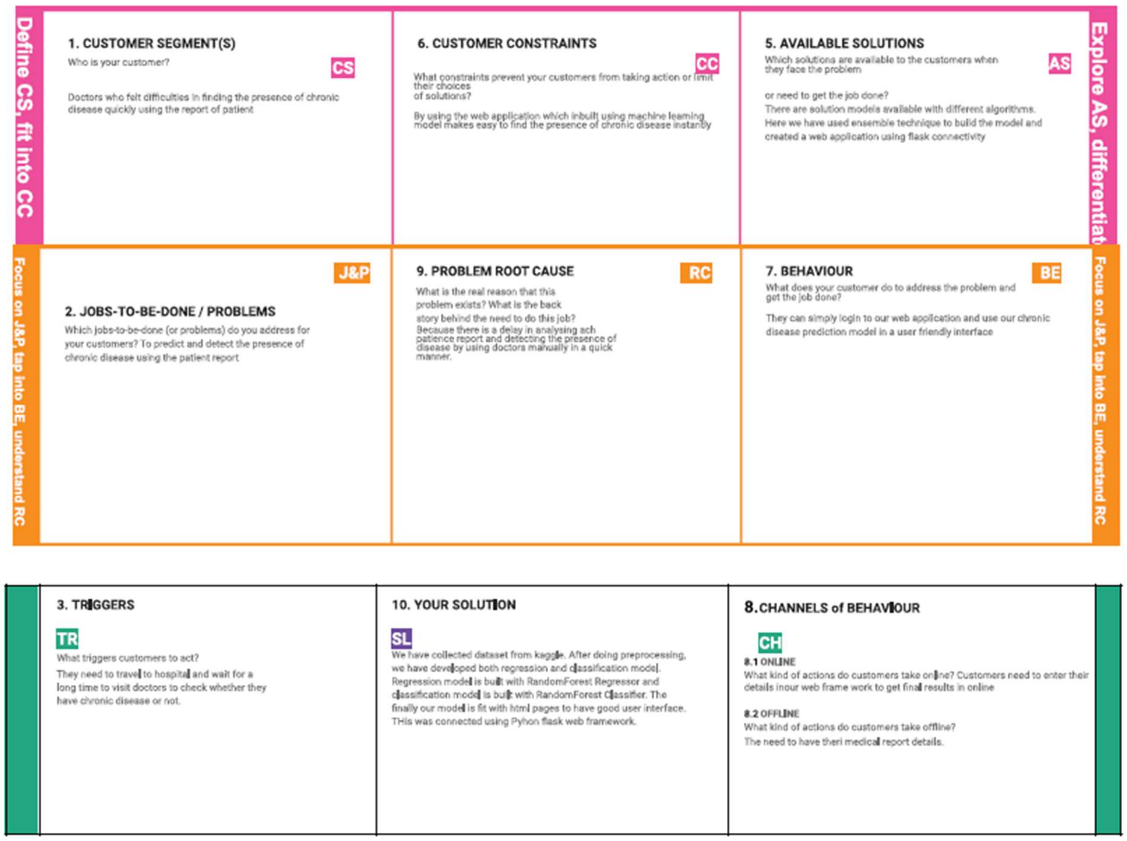
Figure 3.2.3 - Idea Prioritization

3.3 Proposed Solution

S.NO.	PARAMETER	DESCRIPTION
1	Problem Statement (Problem to be solved)	Kidney diseases avert the normal function of the kidney. Early prediction of kidney disease using both classification and regression algorithms are an effective task that can help the doctors to diagnose the disease within a short duration of time.
2	Idea / Solution description	One of the easiest solutions to predict the kidney disease using Machine Learning techniques.
3	Novelty / Uniqueness	This project provides the best accuracy for predicting the kidney disease.
4	Social Impact / Customer Satisfaction	It helps to identify the kidney disease in effective way, reduce the cost and user friendly.
5	Scalability of the Solution	This project can be improved by giving medical suggestion for patients.

Table 3.3.1 - Proposed Solution

3.4 Problem Solution Fit


Define CS, fit into CC
Explore AS, differentiat
Focus on J&P, tap into BE, understand RC
Focus on J&P, tap into BE, understand RC

1. CUSTOMER SEGMENT(S) CS

Who is your customer?

Doctors who felt difficulties in finding the presence of chronic disease quickly using the report of patient

6. CUSTOMER CONSTRAINTS CC

What constraints prevent your customers from taking action or limit their choices
of solutions?

By using the web application which inbuilt using machine learning model makes easy to find the presence of chronic disease instantly

5. AVAILABLE SOLUTIONS AS

Which solutions are available to the customers when they face the problem

or need to get the job done?
There are solution models available with different algorithms.
Here we have used ensemble technique to bulid the model and created a web application using flask connectivity

2. JOBS-TO-BE-DONE / PROBLEMS J&P

Which jobs-to-be-done (or problems) do you address for your customers? To predict and detect the presence of chronic disease using the patient report

9. PROBLEM ROOT CAUSE RC

What is the real reason that this problem exists? What is the back story behind the need to do this job?
Because there is a delay in analysing ach patience report and detecting the presence of disease by using doctors manually in a quick manner.

7. BEHAVIOUR BE

What does your customer do to address the problem and get the job done?

They can simply login to our web application and use our chronic disease prediction model in a user friendly interface

3. TRIGGERS

TR

What triggers customers to act?
They need to travel to hospital and wait for a long time to visit doctors to check whether they have chronic disease or not.

10. YOUR SOLUTION

SL

We have collected dataset from kaggle. After doing preprocessing, we have developed both regression and classification model. Regression model is built with RandomForest Regressor and classification model is built with RandomForest Classifier. The finally our model is fit with html pages to have good user interface. THis was connected using Pyhon flask web framework.

8. CHANNELS of BEHAVIOUR

CH

8.1 ONLINE
What kind of actions do customers take online? Customers need to enter their details inour web frame work to get final results in online

8.2 OFFLINE
What kind of actions do customers take offline?
The need to have theri medical report details.

Figure 3.4.1 - Problem Solution Fit

CHAPTER 4

REQUIREMENT ANALYSIS

4.1 Functional Requirement

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	Home Page	• Chronic Kidney disease description • Information about Test Vitals required for prediction • If new User, REGISTER • If already exist, SIGN
FR-2	User Registration	• Enters Mail ID and other personal details required for Registering
FR-3	User Login	• Uses Mail ID and Password for login
FR-4	Test Vitals Form	• Test Vitals should be entered for prediction
FR-5	Result	• If Positive – Test Result along with the Information about what is to be done next will be displayed. • If Negative – Test result along with preventive measures to prevent themselves from getting chronic kidney disease will be displayed.

Table 4.1.1 - Functional Requirements

4.2 Non-Functional Requirements

Following are the non-functional requirements of the proposed solution

NFR No.	Non-Functional Requirement	Description
NFR-1	Usability	Even Illiterates and people with no understanding of computer/mobile should be able to use the product.
NFR-2	Security	Access permission for particular system information may be changed by systems data administration.
NFR-3	Reliability	The database update process must roll back all related updates when any updates fail.
NFR-4	Performance	The Home-page load time must be no more than 2 seconds for users that access the website using an LTE mobile connection.
NFR-5	Availability	New Model Deployment must not impact home page, test page and result page availability and must not take longer than 1 hour.
NFR-6	Scalability	The website Traffic limit must be scalable enough to support 2000,000 users at a time.

Table 4.2.1 - Non-Functional Requirements

CHAPTER 5
PROJECT DESIGN

5.1 Data flow diagrams

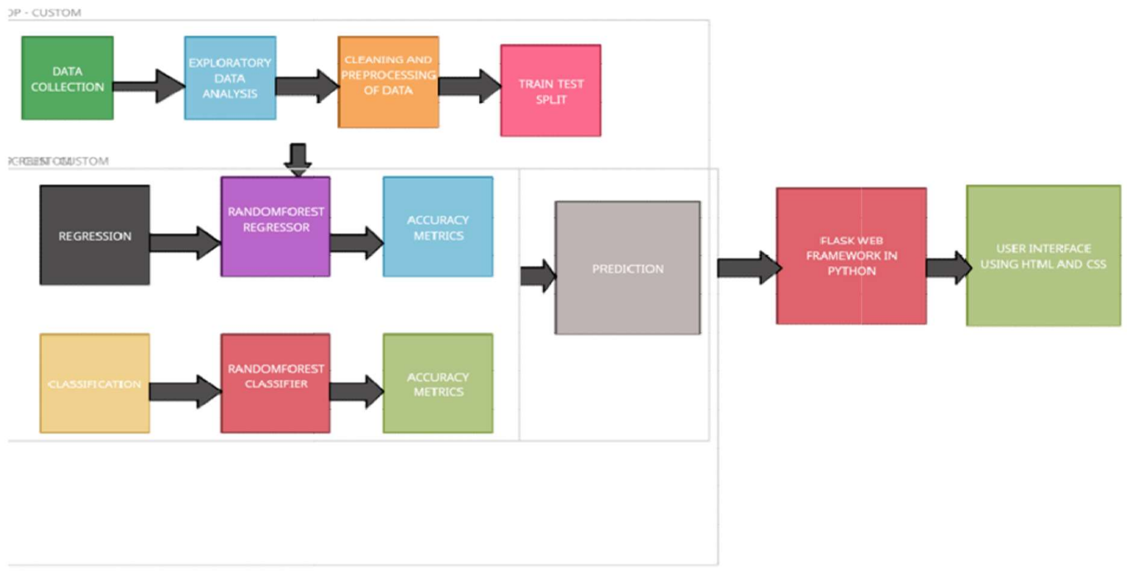


Fig 5.1.1 - Data Flow of Chronic Disease Prediction

1. Medical data of patients is collected from Kaggle.
2. Exploratory data analysis done on the input dataset.
3. Then removal of null values, duplicates and outliers.
4. Then the dependent and independent variable is defined.
5. Train test split is done.
6. Both classification and regression model are built.
7. For Classification, the model is trained with Random Forest Classifier and tested with
8. test dataset.
9. For Regression, the model is trained with Random Forest Regression and tested with test dataset.
10. Then the model is fitted with front end which is developed using HTML, CSS with the help of Python Flask Web Framework.
11. Finally, the output will be predicted for the user input data.

5.2 Solution and Technical Architecture

- The best solution to predict kidney disease using Machine Learning Techniques.
- Early prediction of kidney disease using classification and regression algorithms are an effective task that can help the doctors to diagnose the disease within short duration of time.
- It helps to identify the kidney disease in an effective way and can be improved by giving medical suggestion for patients.

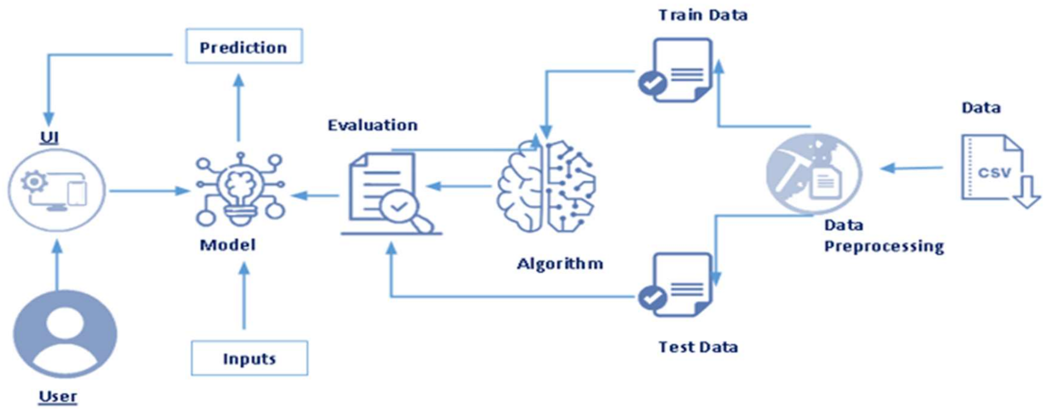


Figure 5.2.1 - Solution Architecture

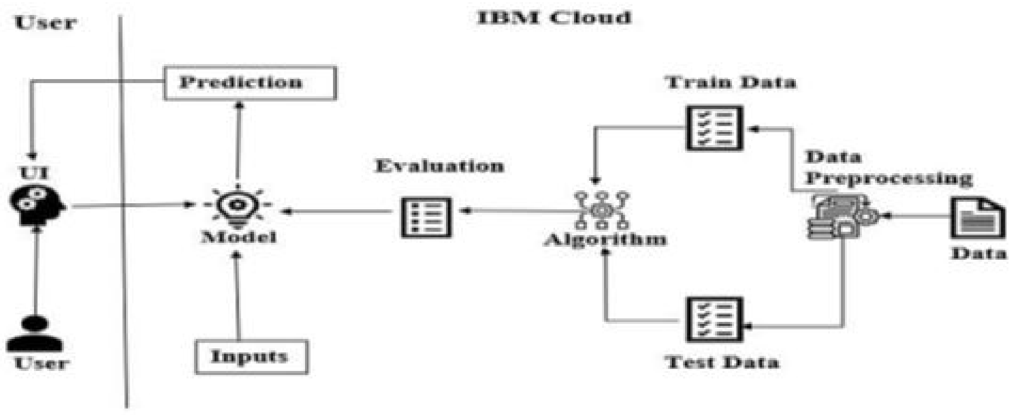


Figure 5.2.2 - Technical Architecture

By the help of this technical architecture we can able to implement the Early Detection of Chronic Kidney Disease.

5.3 User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Web user)	Registration	USN- 1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
	Verification	USN- 2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
	Login	USN- 3	As a user, I can login to the application by entering email and password.	Check whether password and email is correct	High	Sprint-1
	Dashboard	USN- 4	If the email id and password is correct, the user can log in to the application otherwise it shows ‘incorrect password or Id’.	View the dashboard of user who is log in	High	Sprint-1

Customer Care Executive	Help	USN- 5	If the user faces any issues, he/she can report it to our mail id.	Report option will be available in web app	High	Sprint-2
Administrator	Verification	USN- 6	Administrator or also has unique Id and password to login. He has additional users to organize the users of this web app	Check whether password and email are correct	High	Sprint-3

Table 5.3.1 - User Story

CHAPTER 6

PROJECT PLANNING AND SCHEDULING

6.1 Sprint Planning & Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	User Registration	USN-1	As a user, I can register for the application by entering my name, mobile number, email, password, and confirming my password.	10	High	Monishsurya SM Ranjith Kumar P Deepakraj k Naveen Kumar P
Sprint-1		USN-2	As a user, I can register for the application through Gmail.	5	Medium	Monishsurya SM Ranjith Kumar P Deepakraj k Naveen Kumar P
Sprint-1	Login	USN-3	As a user, I will receive confirmation email once I have registered for the application.	10	High	Monishsurya SM Ranjith Kumar P Deepakraj k Naveen Kumar P
Sprint-2		USN-4	As a user, I will receive confirmation OTP to verify the identity.	5	Medium	Monishsurya SM Ranjith Kumar P Deepakraj k Naveen Kumar P
Sprint-2	Data Collection	USN-5	As a user, I will enter the input data for disease prediction in the form.	10	High	Monishsurya SM Ranjith Kumar P Deepakraj k Naveen Kumar P

Sprint-3	Provide output to the user	USN-6	As a user, I will get the result of disease prediction in the dashboard.	10	High	Monishsurya SM Ranjith Kumar P Deepakraj k Naveen Kumar P
Sprint-3	Data Analysis	USN-7	As the admin, I will develop modules to pre-process and store the data.	10	High	Monishsurya SM Ranjith Kumar P Deepakraj k Naveen Kumar P
Sprint-4	Prediction of disease	USN-8	As the admin, I will build a Machine Learning model to predict the disease.	10	High	Monishsurya SM Ranjith Kumar P Deepakraj k Naveen Kumar P
Sprint-4	Final Delivery	USN-9	Deploy the application in IBM cloud and make it available for use.	10	High	Monishsurya SM Ranjith Kumar P Deepakraj k Naveen Kumar P

Table 6.1.1 - Sprint Planning & Estimation

Project Tracker, Velocity & Burndown Chart:

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24-10-2022	29-10-2022	20	29-10-2022
Sprint-2	20	6 Days	31-10-2022	05-11-2022	20	05-11-2022
Sprint-3	20	6 Days	07-11-2022	12-11-2022	20	12-11-2022
Sprint-4	20	6 Days	14-11-2022	19-11-2022	20	19-11-2022

Table 6.1.2 - Sprint Delivery Schedule

Velocity:

Imagine we have a 6-day sprint duration, and the velocity of the team is 20 (points per print). Let’s calculate the team’s average velocity (AV) per iteration unit (story points per day)

AV = Sprint Duration / Velocity = 20/6 = 3.33

6.2 Project Delivery Schedule

TITLE	DESCRIPTION	DATE
Literature Survey & Information Gathering	Literature survey on the selected project & gathering information by referring the technical papers, research publications, journals etc.	08-Sep-2022
Prepare Empathy Map	Prepare Empathy Map Canvas to capture the user Pains & Gains, prepare list of problem Statements that are to be solved by this project.	08-Sep-2022
Ideation	List the ideas by organizing brainstorming session and prioritize the top 3 ideas based on the feasibility & importance.	15-Sep-2022
Proposed Solution	Prepare the proposed solution document, which includes novelty, feasibility of idea, revenue model, social impact, scalability of solution, etc.	21-Sep-2022
Problem Solution Fit	Prepare problem - solution fit document.	30-Sep-2022
Solution Architecture	Prepare solution architecture document.	28-Sep-2022
Customer Journey	Prepare the customer journey maps to understand the user interactions & experiences with the application (entry to exit).	06-Oct-2022
Functional Requirement	Prepare the functional requirement document.	11-Oct-2022
Data Flow Diagrams	Draw the data flow diagrams and submit for review.	13-Oct-2022

Technology Architecture	Prepare the technology architecture	14-Oct-2022
Prepare Milestone & Activity List	Prepare the milestones &activity list of the project.	19-Oct-2022
Project Development - Delivery of Sprint-1, 2, 3 &4	Develop & submit the developed code by testing it.	IN PROGRESS...

Table 6.2.1 - Project Delivery Schedule

6.3 Reports from Jira

Burndown Chart

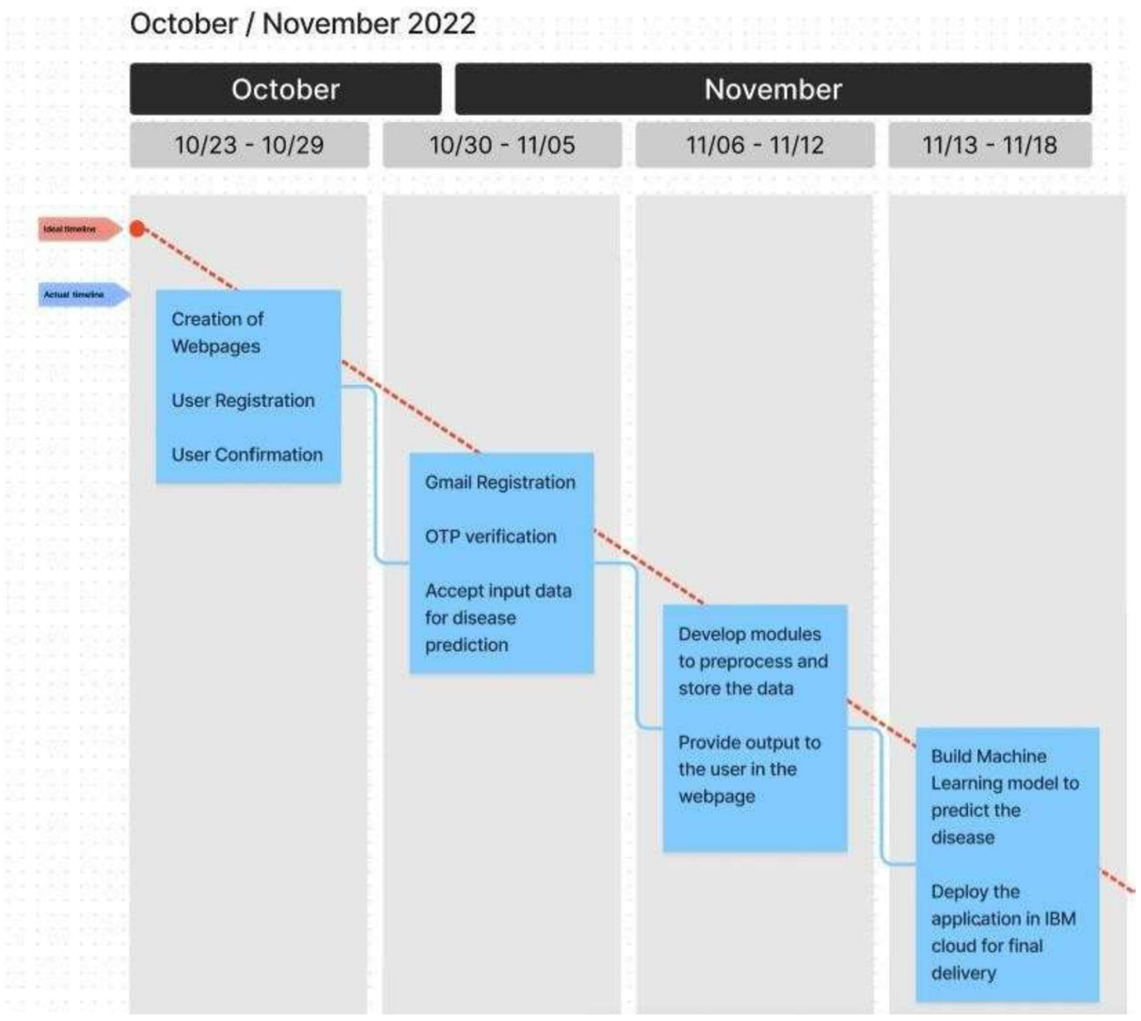


Figure 6.3.1 - Burndown Chart

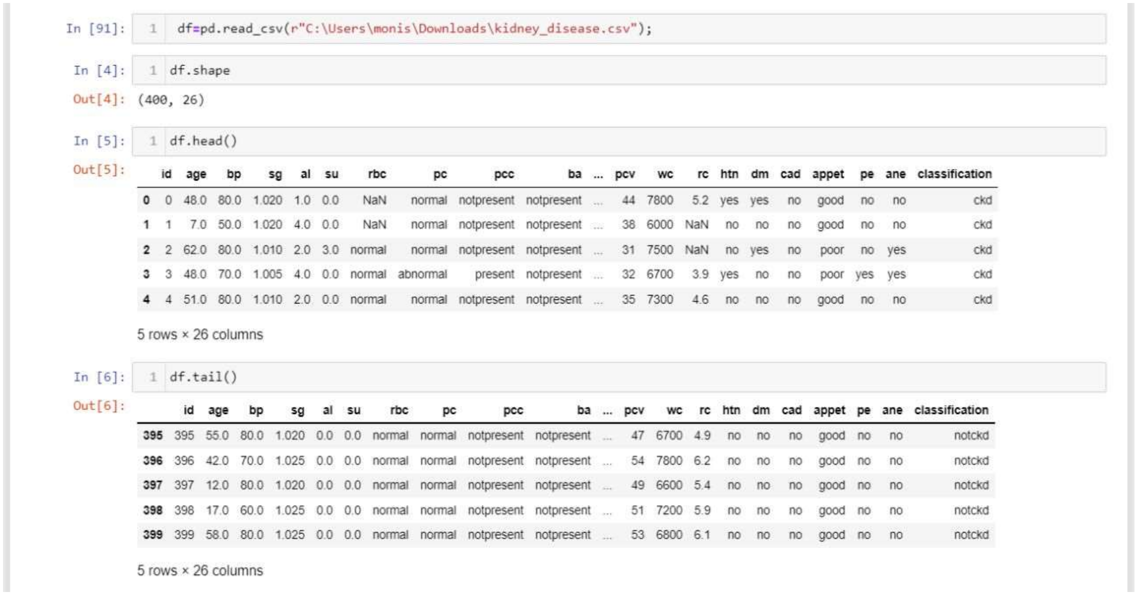
CHAPTER 7
CODING AND SOLUTIONING

7.1 Feature 1

Different types of python libraries such as pandas, Sklearn, NumPy, matplotlib are used for processing the algorithms. Using exploration data analysis technique data was analyses in junketeer notebook.10-fold cross validation technique is used for spitting the data set into training and testing data. Then using random forest algorithm dataset was processed.

Collection of Dataset

For the proposed study dataset was taken from Kaggle site. Then it was downloaded in excel file using comma separated format. Data has processed by python programming using Jupiter notebook. The data set contains 401 sample instances. The dataset contains 26 clinical features.



```
In [91]: df=pd.read_csv(r"C:\Users\monis\Downloads\kidney_disease.csv");

In [4]: df.shape
Out[4]: (400, 26)

In [5]: df.head()
Out[5]:
```

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

5 rows × 26 columns

```
In [6]: df.tail()
Out[6]:
```

	id	age	bp	sg	al	su	rbc	pc	pcc	ba	...	pcv	wc	rc	htn	dm	cad	appet	pe	ane	classification
395	395	55.0	80.0	1.020	0.0	0.0	normal	normal	notpresent	notpresent	...	47	6700	4.9	no	no	no	good	no	no	notckd
396	396	42.0	70.0	1.025	0.0	0.0	normal	normal	notpresent	notpresent	...	54	7800	6.2	no	no	no	good	no	no	notckd
397	397	12.0	80.0	1.020	0.0	0.0	normal	normal	notpresent	notpresent	...	49	6600	5.4	no	no	no	good	no	no	notckd
398	398	17.0	60.0	1.025	0.0	0.0	normal	normal	notpresent	notpresent	...	51	7200	5.9	no	no	no	good	no	no	notckd
399	399	58.0	80.0	1.025	0.0	0.0	normal	normal	notpresent	notpresent	...	53	6800	6.1	no	no	no	good	no	no	notckd

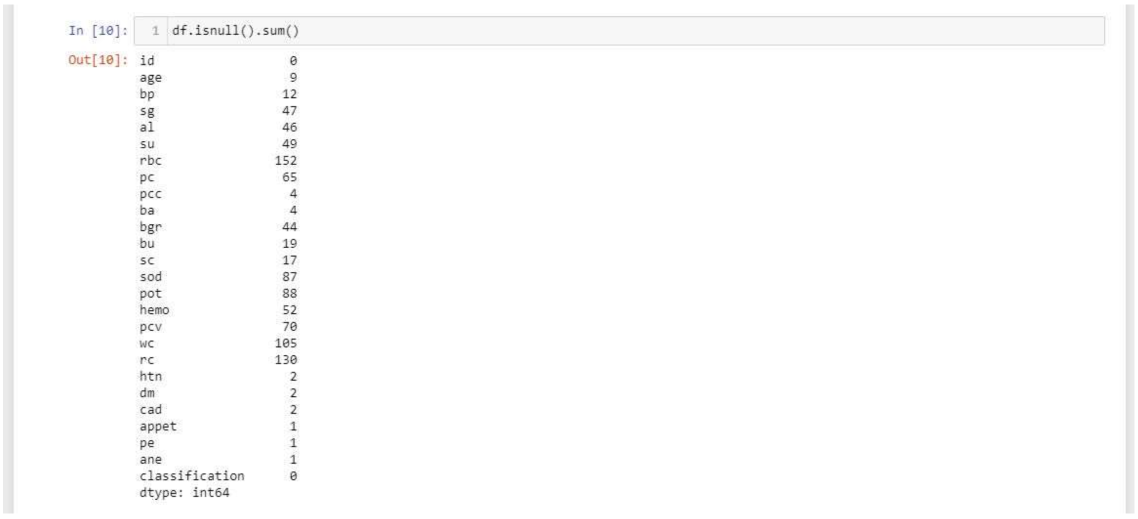
5 rows × 26 columns

Figure 7.1.1 - Collection of Dataset

Preprocessing - Data cleaning

Checking Null Entries

The most important step in EDA involving removing duplicate rows/columns, filling the void entries with values like mean/median of the data, dropping various values, removing null entries Here we have check for null values and drop the entries which contains null values as the percentage of null values in dataset is very less.

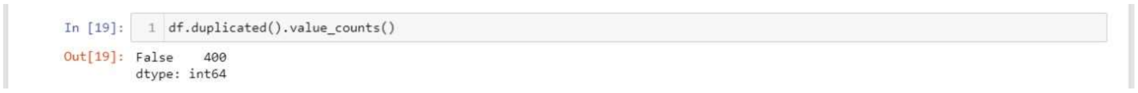


```
In [10]:  df.isnull().sum()

Out[10]: id                 0
         age                9
         bp                12
         sg                47
         al                46
         su                49
         rbc              152
         pc                65
         pcc                4
         ba                 4
         bgr               44
         bu                19
         sc                17
         sod               87
         pot               88
         hemo              52
         pcv               70
         wc               105
         rc               130
         htn                2
         dm                 2
         cad                2
         appet              1
         pe                 1
         ane                1
         classification     0
         dtype: int64
```

Figure 7.1.2 - Checking Null Entries

Checking Duplicates

In the dataset, there is no duplicate entries.

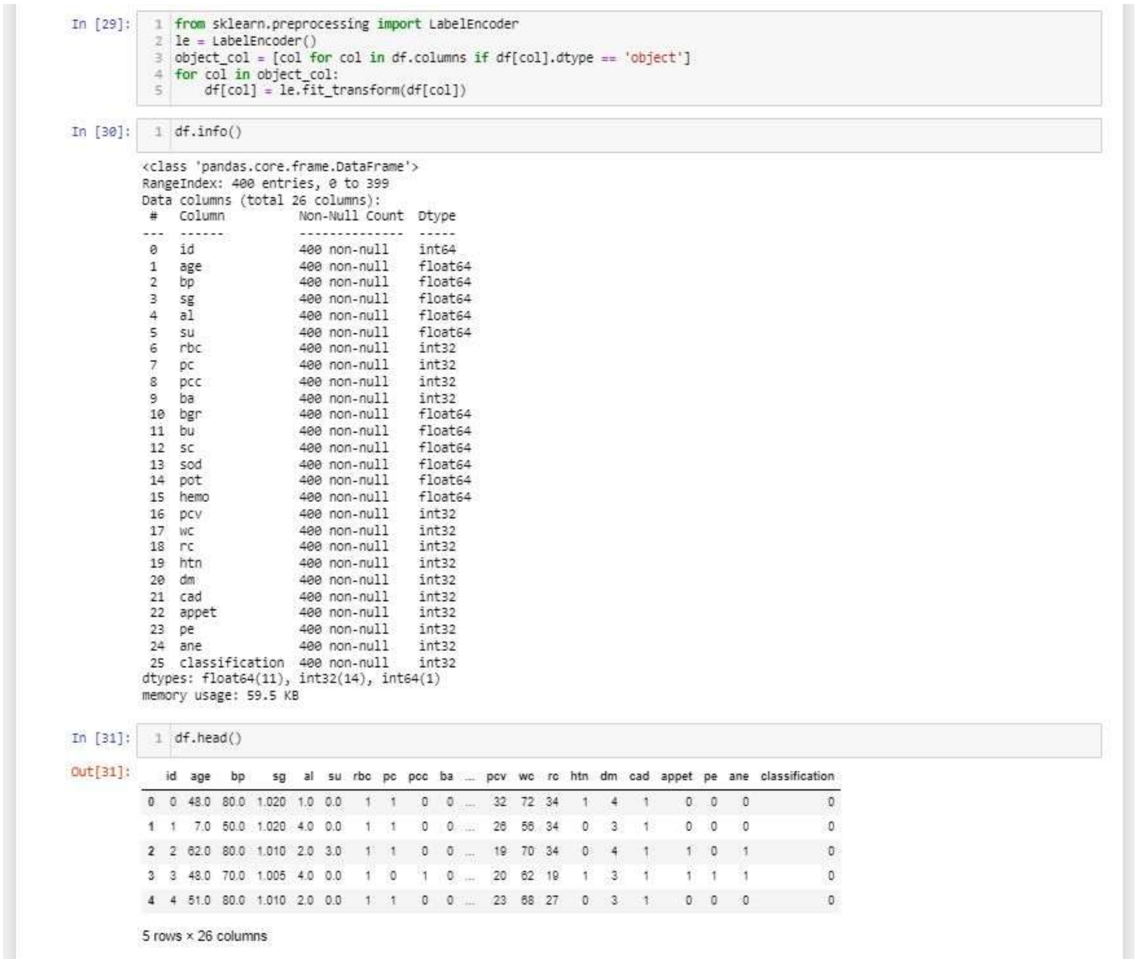


```
In [19]:  df.duplicated().value_counts()

Out[19]: False    400
         dtype: int64
```

Figure 7.1.3 - Checking Duplicates

Encoding:

All the categorical columns('htn','dm','cad','pe','ane', 'rbc','pc', 'pcc','ba', 'appet', classification') in dataset is converted into numerical.



In [29]:
```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
object_col = [col for col in df.columns if df[col].dtype == 'object']
for col in object_col:
    df[col] = le.fit_transform(df[col])
```

In [30]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 400 entries, 0 to 399
Data columns (total 26 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   id              400 non-null    int64
 1   age             400 non-null    float64
 2   bp              400 non-null    float64
 3   sg              400 non-null    float64
 4   al              400 non-null    float64
 5   su              400 non-null    float64
 6   rbc             400 non-null    int32
 7   pc              400 non-null    int32
 8   pcc             400 non-null    int32
 9   ba              400 non-null    int32
 10  bgr             400 non-null    float64
 11  bu              400 non-null    float64
 12  sc              400 non-null    float64
 13  sod             400 non-null    float64
 14  pot             400 non-null    float64
 15  hemo            400 non-null    float64
 16  pcv             400 non-null    int32
 17  wc              400 non-null    int32
 18  rc              400 non-null    int32
 19  htn             400 non-null    int32
 20  dm              400 non-null    int32
 21  cad             400 non-null    int32
 22  appet           400 non-null    int32
 23  pe              400 non-null    int32
 24  ane             400 non-null    int32
 25  classification  400 non-null    int32
dtypes: float64(11), int32(14), int64(1)
memory usage: 59.5 KB
```

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

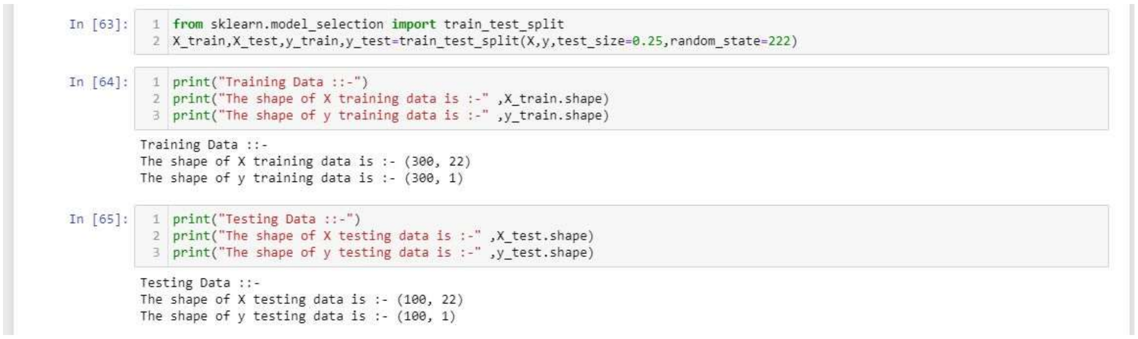
Out[31]:

	id	age	bp	sg	al	su	rbc	pc	pcc	ba	...	pcv	wc	rc	htn	dm	cad	appet	pe	ane	classification
0	0	48.0	80.0	1.020	1.0	0.0	1	1	0	0	...	32	72	34	1	4	1	0	0	0	0
1	1	7.0	50.0	1.020	4.0	0.0	1	1	0	0	...	26	56	34	0	3	1	0	0	0	0
2	2	62.0	80.0	1.010	2.0	3.0	1	1	0	0	...	19	70	34	0	4	1	1	0	1	0
3	3	48.0	70.0	1.005	4.0	0.0	1	0	1	0	...	20	62	19	1	3	1	1	1	1	0
4	4	51.0	80.0	1.010	2.0	0.0	1	1	0	0	...	23	68	27	0	3	1	0	0	0	0

5 rows × 26 columns

Figure 7.1.4 - Cleaning and preprocessing the data

Split of dependent and Independent Variables:



```
In [63]:  from sklearn.model_selection import train_test_split
          X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.25,random_state=222)

In [64]:  print("Training Data ::-")
          print("The shape of X training data is :-" ,X_train.shape)
          print("The shape of y training data is :-" ,y_train.shape)

          Training Data ::-
          The shape of X training data is :- (300, 22)
          The shape of y training data is :- (300, 1)

In [65]:  print("Testing Data ::-")
          print("The shape of X testing data is :-" ,X_test.shape)
          print("The shape of y testing data is :-" ,y_test.shape)

          Testing Data ::-
          The shape of X testing data is :- (100, 22)
          The shape of y testing data is :- (100, 1)
```

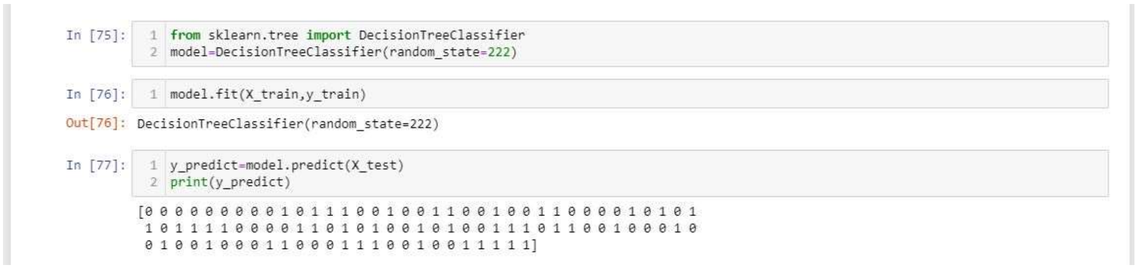
Figure 7.1.5 Split of dependent and Independent Variables

7.2 Feature 2

Both Classification and Regression models are built for this use case.

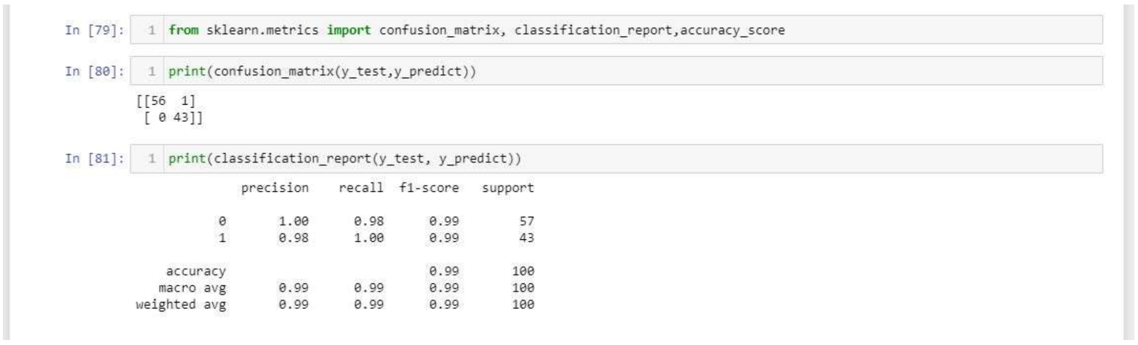
For classification:

Decision Tree is a supervised learning algorithm which is used for both classification and regression problems. Decision tree classifier is mostly used for the classification problems. From decision tree, get set of rules for classifying the problem.



```
In [75]:  from sklearn.tree import DecisionTreeClassifier
          model=DecisionTreeClassifier(random_state=222)

In [76]:  model.fit(X_train,y_train)

Out[76]: DecisionTreeClassifier(random_state=222)

In [77]:  y_predict=model.predict(X_test)
          print(y_predict)

          [0 0 0 0 0 0 0 0 0 1 0 1 1 1 0 0 1 0 0 1 1 0 0 1 0 0 1 1 0 0 0 0 1 0 1 0 1
           1 0 1 1 1 1 0 0 0 0 1 1 0 1 0 1 0 0 1 0 1 0 0 1 1 1 0 1 1 0 0 1 0 0 0 1 0
           0 1 0 0 1 0 0 0 1 1 0 0 0 1 1 1 0 0 1 0 0 1 1 1 1 1]
```

Figure 7.2.1 Choosing Parameters

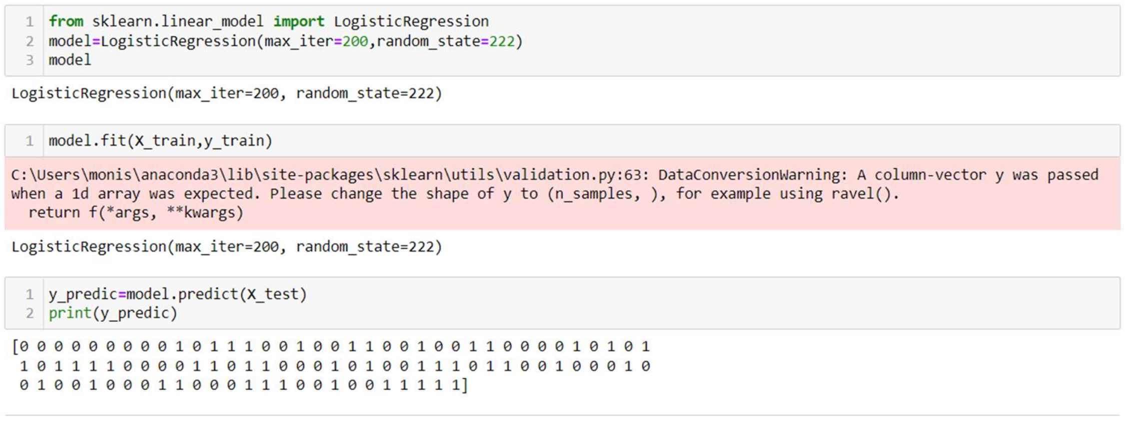


```
In [79]:  from sklearn.metrics import confusion_matrix, classification_report,accuracy_score

In [80]:  print(confusion_matrix(y_test,y_predict))

[[56  1]
 [ 0 43]]

In [81]:  print(classification_report(y_test, y_predict))

              precision    recall  f1-score   support

           0       1.00      0.98      0.99        57
           1       0.98      1.00      0.99        43

    accuracy                           0.99       100
   macro avg       0.99      0.99      0.99       100
weighted avg       0.99      0.99      0.99       100
```

Figure 7.2.2 - Choosing Parameters

For Regression:

Logistic Regression is an supervised learning algorithm which is used for the solving classification problems.

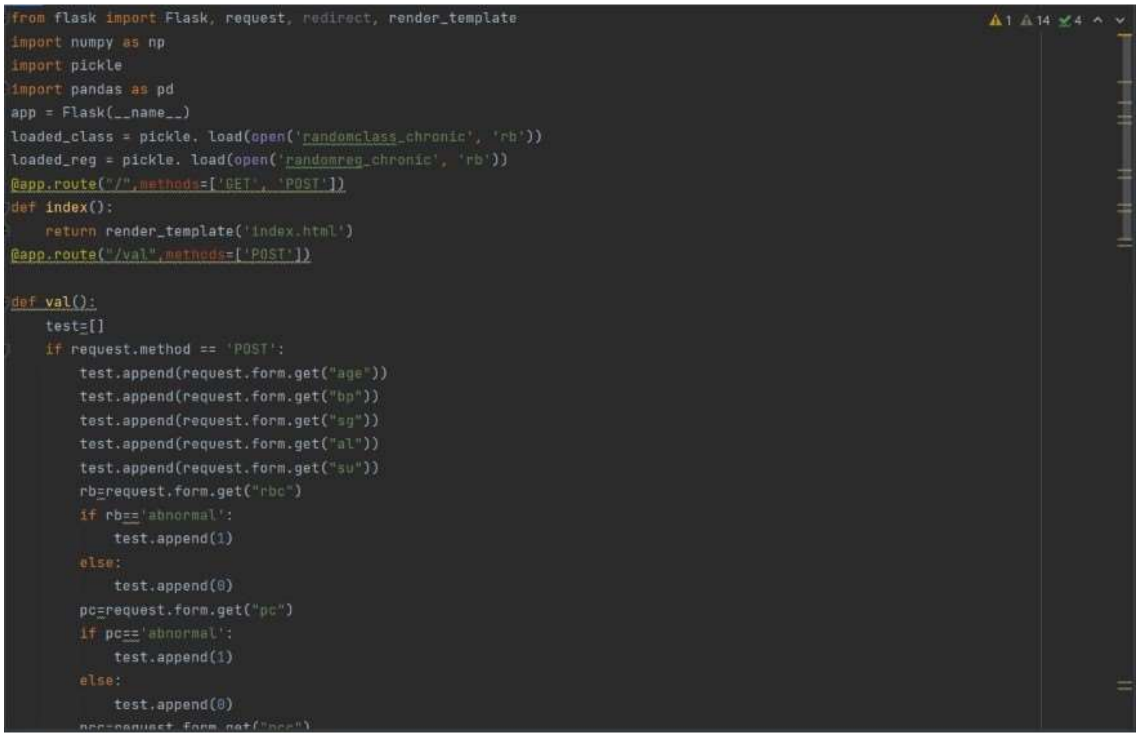


```
from sklearn.linear_model import LogisticRegression
model=LogisticRegression(max_iter=200,random_state=222)
model

LogisticRegression(max_iter=200, random_state=222)

model.fit(X_train,y_train)

C:\Users\monis\anaconda3\lib\site-packages\sklearn\utils\validation.py:63: DataConversionWarning: A column-vector y was passed
when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
  return f(*args, **kwargs)

LogisticRegression(max_iter=200, random_state=222)

y_predic=model.predict(X_test)
print(y_predic)

[0 0 0 0 0 0 0 0 0 1 0 1 1 1 0 0 1 0 0 1 1 0 0 1 0 0 1 1 0 0 0 0 1 0 1 0 1
 1 0 1 1 1 1 0 0 0 0 1 1 0 1 1 0 0 0 1 0 1 0 0 1 1 1 0 1 1 0 0 1 0 0 0 1 0
 0 1 0 0 1 0 0 0 1 1 0 0 0 1 1 1 0 0 1 0 0 1 1 1 1 1]
```

Figure 7.2.3 Logistic Regression

Flask Connectivity

The Backend Machine Learning model code is connect with HTML code by using python Flask Web Framework.



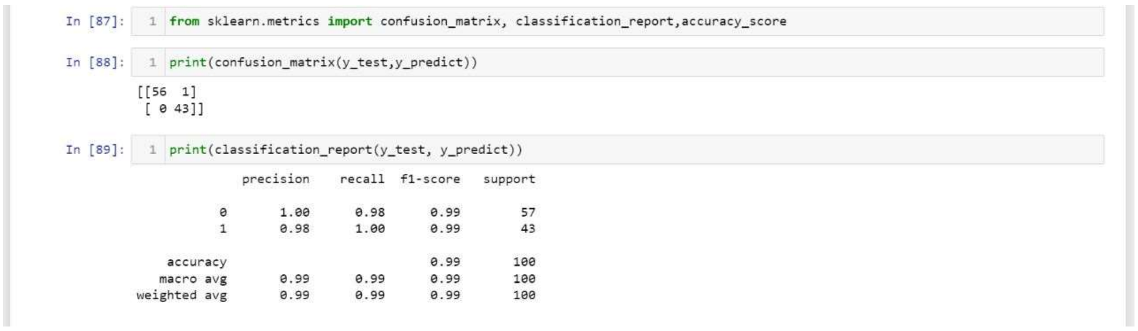
```python
from flask import Flask, request, redirect, render_template
import numpy as np
import pickle
import pandas as pd
app = Flask(__name__)
loaded_class = pickle. load(open('randomclass_chronic', 'rb'))
loaded_reg = pickle. load(open('randomreg_chronic', 'rb'))
@app.route("/",methods=['GET', 'POST'])
def index():
    return render_template('index.html')
@app.route("/val",methods=['POST'])

def val():
    test=[]
    if request.method == 'POST':
        test.append(request.form.get("age"))
        test.append(request.form.get("bp"))
        test.append(request.form.get("sg"))
        test.append(request.form.get("al"))
        test.append(request.form.get("su"))
        rb=request.form.get("rbc")
        if rb=='abnormal':
            test.append(1)
        else:
            test.append(0)
        pc=request.form.get("pc")
        if pc=='abnormal':
            test.append(1)
        else:
            test.append(0)
        pcc=request.form.get("pcc")
```

Figure 7.2.4 - Flask connectivity

CHAPTER 8

TESTING

8.1 Test Cases



```
In [87]:  from sklearn.metrics import confusion_matrix, classification_report,accuracy_score

In [88]:  print(confusion_matrix(y_test,y_predict))

[[56  1]
 [ 0 43]]

In [89]:  print(classification_report(y_test, y_predict))

              precision    recall  f1-score   support

           0       1.00      0.98      0.99        57
           1       0.98      1.00      0.99        43

    accuracy                           0.99       100
   macro avg       0.99      0.99      0.99       100
weighted avg       0.99      0.99      0.99       100
```

Figure 8.1.1 Test Case

8.2 User Acceptance Testing

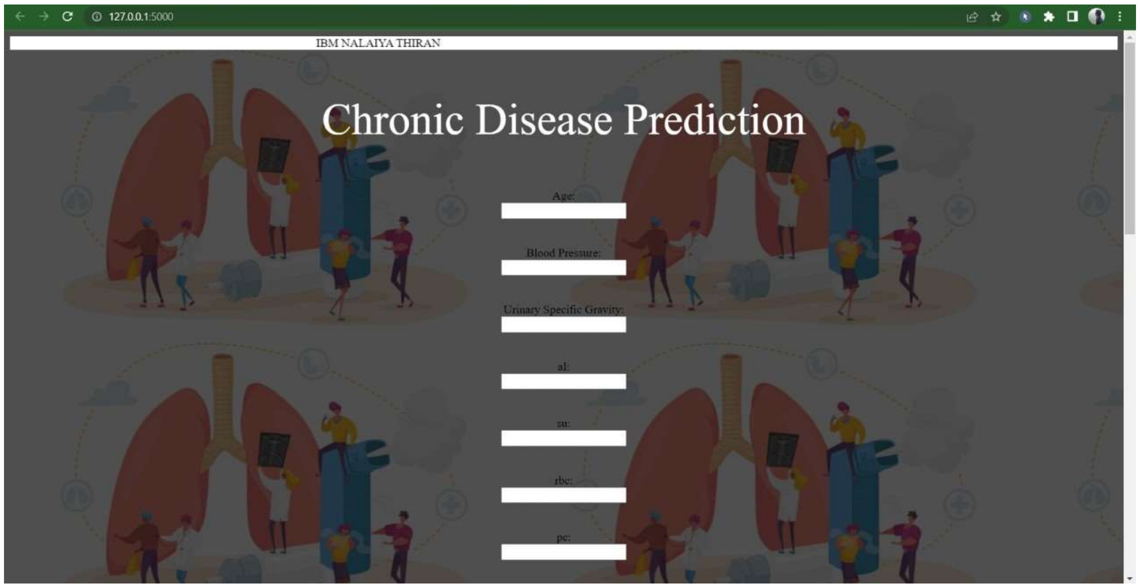
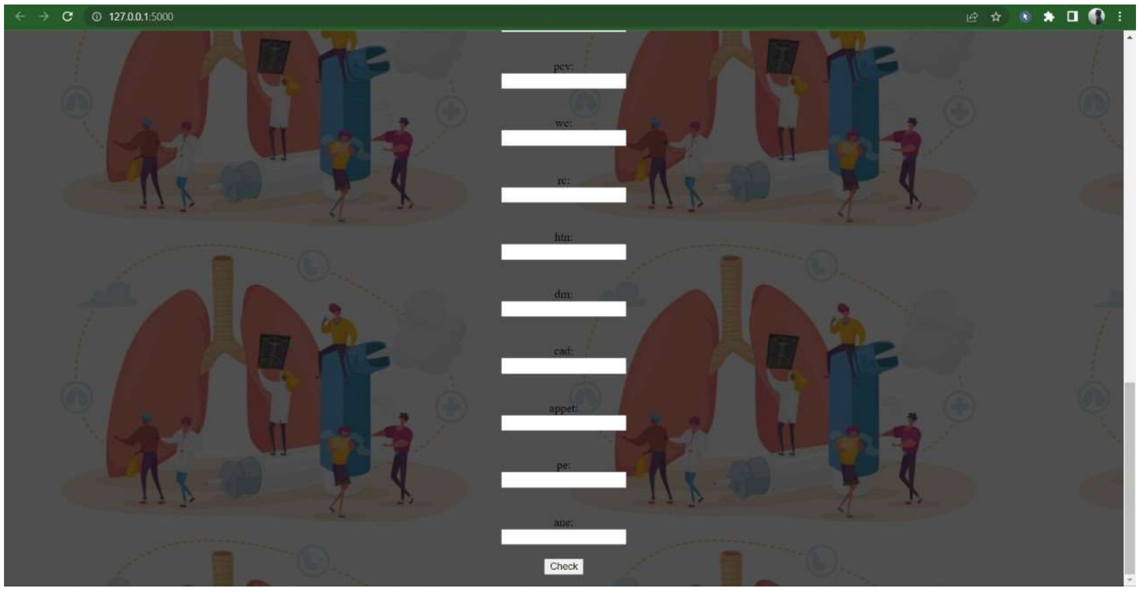


Figure 8.1.2 – Web Page View



127.0.0.1:5000

pcv:

wc:

rc:

htn:

dm:

cad:

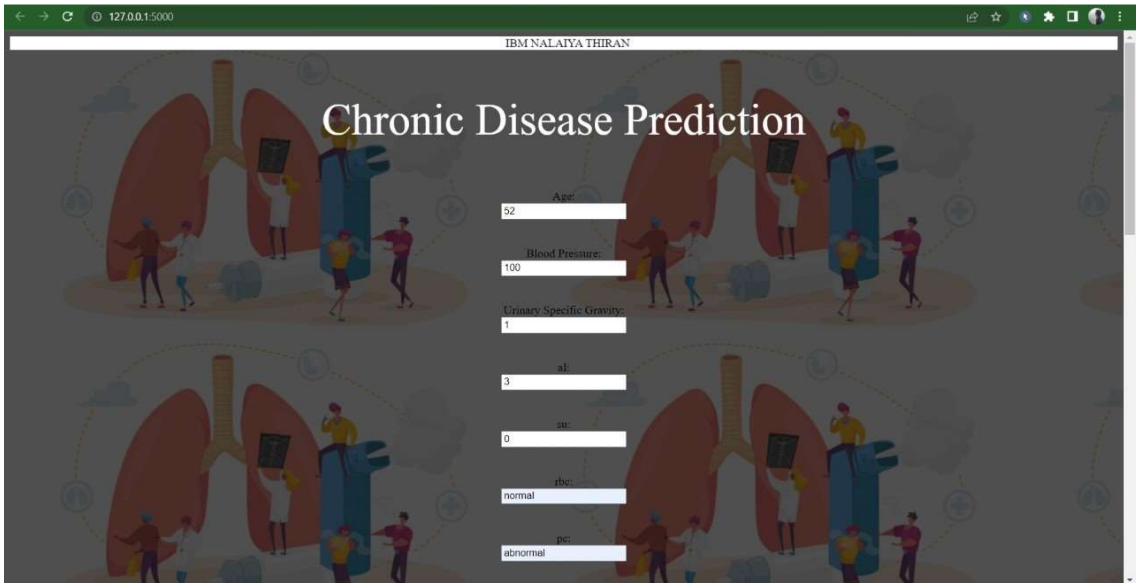
appet:

pe:

ane:

Check

Figure 8.1.3 - Web Page View



127.0.0.1:5000

IBM NALAIYA THIRAN

Chronic Disease Prediction

Age: 52

Blood Pressure: 100

Urinary Specific Gravity: 1

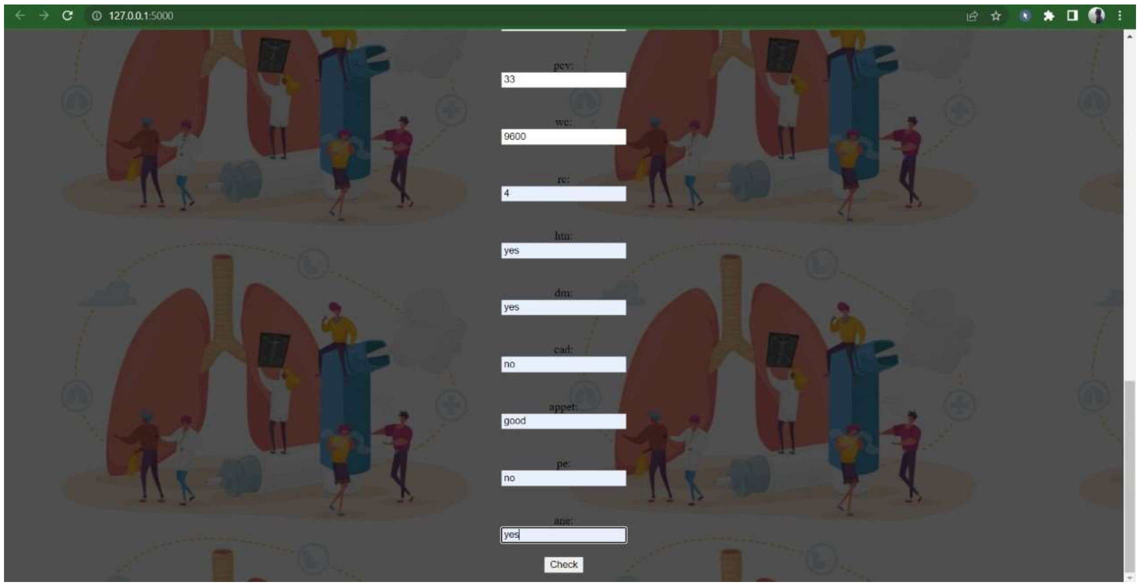
al: 3

su: 0

rbc: normal

pc: abnormal

Figure 8.1.4 - Entering value in the web page



pcv:
33

wc:
9600

rc:
4

htn:
yes

dm:
yes

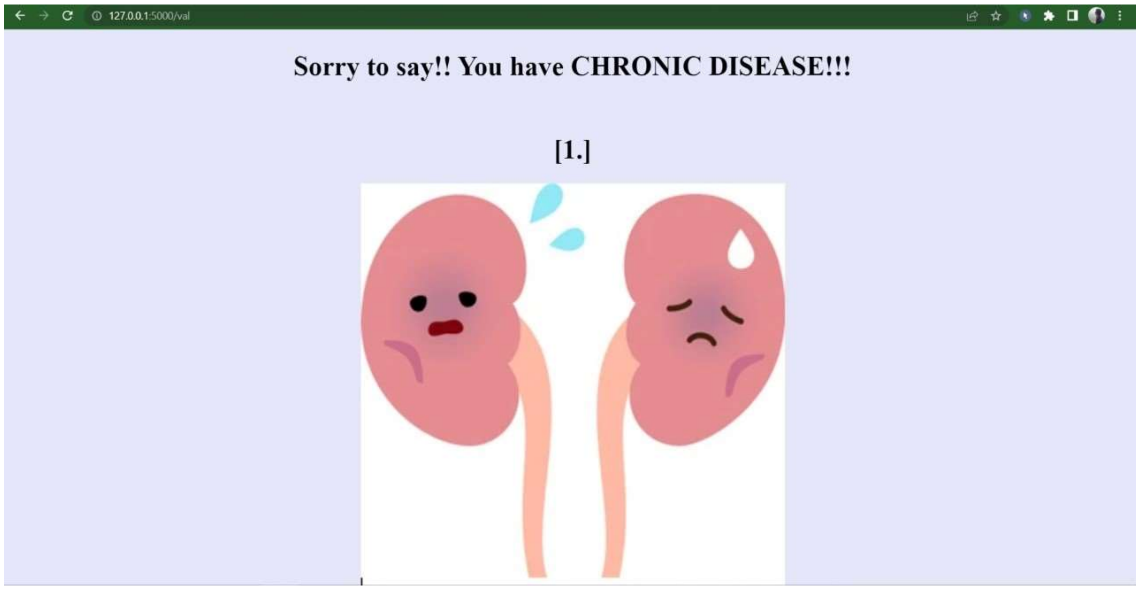
cad:
no

appet:
good

pe:
no

ane:
yes

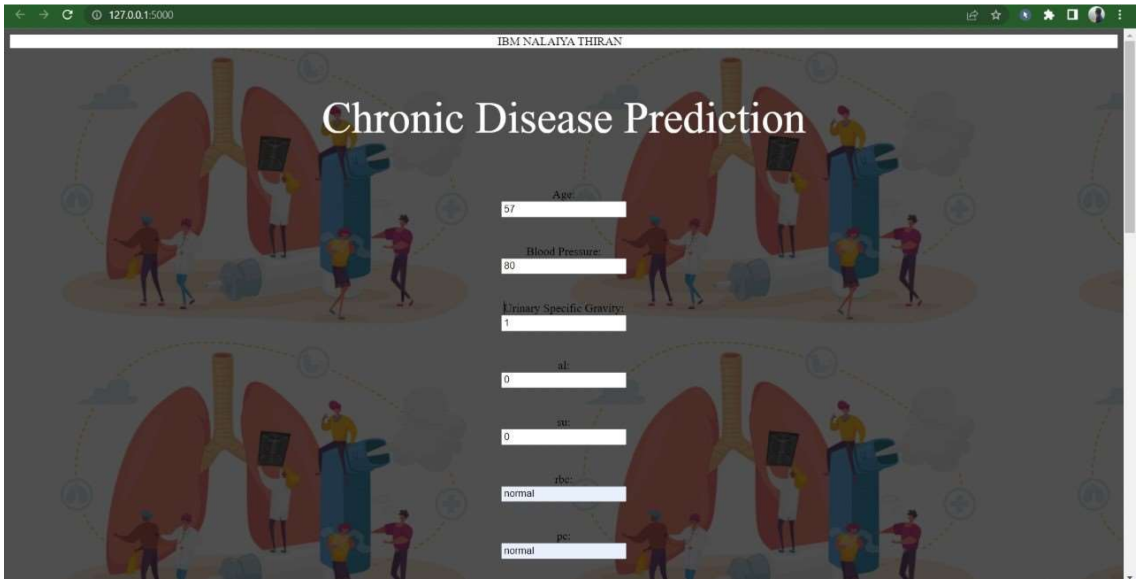
Figure 8.1.5 - Entering value in the web page



Sorry to say!! You have CHRONIC DISEASE!!!

[1.]

Figure 8.1.6 - Result with person having CKD


127.0.0.1:5000

IBM NALAIYA THIRAN

Chronic Disease Prediction

Age: 57

Blood Pressure: 80

Urinary Specific Gravity: 1

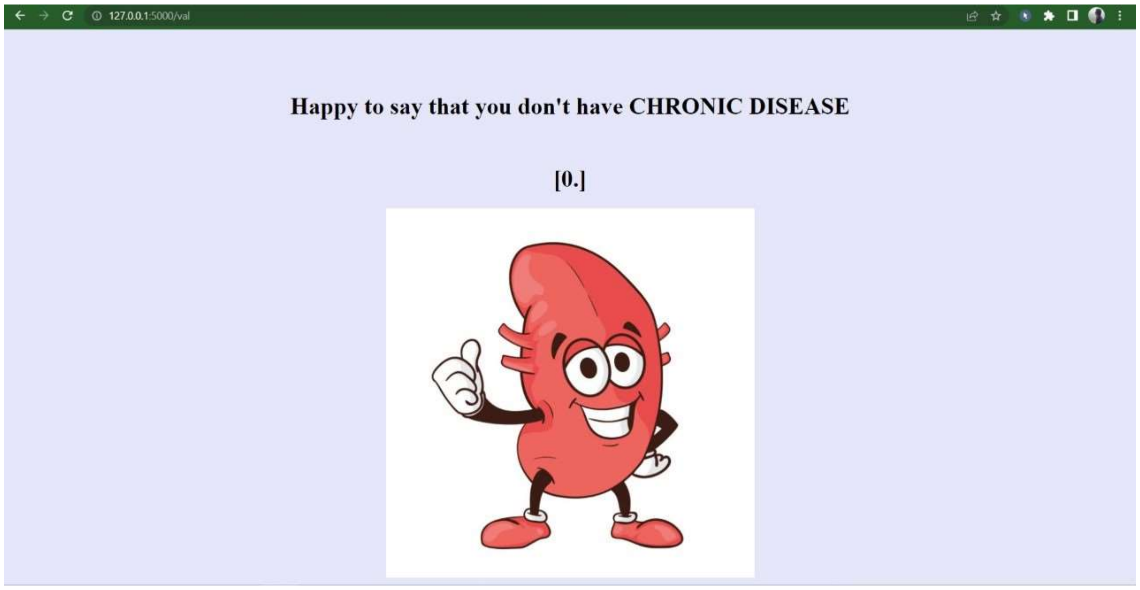
al: 0

su: 0

rbc: normal

pc: normal

Figure 8.1.7 - Entering value in the web page


127.0.0.1:5000/val

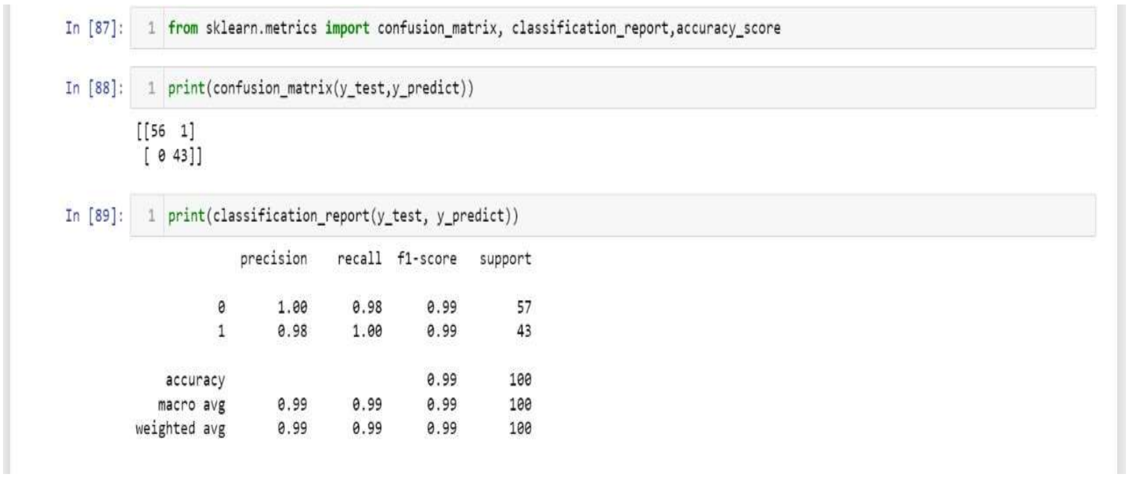
Happy to say that you don't have CHRONIC DISEASE

[0.]

Figure 8.1.8 - - Result with person not having CKD

CHAPTER 9
RESULTS

9.1 Performance Metrics



```
In [87]:  from sklearn.metrics import confusion_matrix, classification_report,accuracy_score

In [88]:  print(confusion_matrix(y_test,y_predict))

[[56  1]
 [ 0 43]]

In [89]:  print(classification_report(y_test, y_predict))

              precision    recall  f1-score   support

           0       1.00      0.98      0.99        57
           1       0.98      1.00      0.99        43

    accuracy                           0.99       100
   macro avg       0.99      0.99      0.99       100
weighted avg       0.99      0.99      0.99       100
```

Figure 9.1.1 - Performance Metrics

One can use following execution measures for the request and figure of imperfection slanted module as shown by his/her own need.

Confusion Matrix: The confusion matrix is used to measure the introduction of two class issue for the given instructive record. The right corner to corner parts TP(True positive)andTN (True Negative) adequately describe instances similarly as FP (false positive) and FN (false negative) wrongly request instances. Confusion Matrix correctly classify instance TP+TN incorrectly classify instances.

1. True positive simply the positive liver tuples that were precisely named by the classifier,
2. True negatives are the negative liver tuples that were precisely set apart by theclassifier.
3. False positives are the negative liver tuples that were erroneously set apart as positivetuples
4. False negatives are the positive liver tuples that were incorrectly stamped negative tuples.

CHAPTER 10
ADVANTAGES AND DISADVANTAGES

10.1 Advantages

The early detection of CKD allows patients to receive timely treatment, slowing the disease's progression. Due to its rapid recognition performance and accuracy, machine learning models can effectively assist physicians in achieving this goal. Chronic kidney disease(CKD) is a type of kidney disease in which there is gradual loss of kidney function over a period of months to years, Initially, there are generally no symptoms; later, symptoms may include leg swelling, feeling tired, vomiting, loss of appetite, and confusion. Complications include an increased risk of heart disease, high blood pressure, bone disease, and anaemia. CKD is associated with a decrease in kidney function related to age and is accelerated in hypertension, diabetes, obesity, and primary kidney disorders. CKD is a global health problem with a high morbidity and mortality rate, and it induces other diseases. As there are no obvious symptoms during the early stages of CKD, patients often do not notice the disease, this being the main feature, eventually leading to a complete loss of kidney function. Early detection of CKD allows patients to receive timely treatment to improve the progression of this disease. As it has been proposed in the objectives of the work, the aim is to develop an automatic learning model for the prediction in the diagnosis of CKD and to contribute to the reduction of significant complications in the disease such as dialysis processes, kidney transplantation, or reaching death. The main criterion of success for this project, with the help of machine learning, is to identify the behaviours or behaviour patterns in the initial stages of CKD to improve the quality of life of patients.

10.2 Disadvantages

The idea for the approach of this project arises from the current situation regarding the increase in the confirmatory diagnosis of kidney, and lack of treatment or the user's ignorance of its pathologies leads to irreversible kidney failure in the final stages of disease, such as dialysis for life, financially affecting the health system, as it is a costly treatment that generates the most significant amount of absorption of the resources available for health. This could be reduced by using tools such as machine learning to classify from the initial stages. Although the application of machine learning in healthcare and other areas is favourable, the field of kidney disease has not yet exploited its full potential.

CHAPTER 11
CONCLUSION

Chronic Kidney Disease (CKD) or chronic renal disease has become a major issue with a steady growth rate. A person can only survive without kidneys for an average time of 18 days, which makes a huge demand for a kidney transplant and Dialysis. It is important to have effective methods for early prediction of CKD. Machine learning methods are effective in CKD prediction. This work proposes a workflow to predict CKD status based on clinical data, incorporating data prepossessing, a missing value handling method with collaborative filtering and attributes selection. Out of the 11 machine learning methods considered, the extra tree classifier and random forest classifier are shown to result in the highest accuracy and minimal bias to the attributes. The research also considers the practical aspects of data collection and highlights the importance of incorporating domain knowledge when using machine learning for CKD status prediction.

CHAPTER 12
FUTURE SCOPE

The increasing rate of CKD has placed a large negative impact on individuals' lives. Advanced ML technology has made the early detection of CKD easier and more accurate. Doctors and medical care professionals have used ML algorithms in the effective diagnosis of CKD. However, there is very little research on the detection of secondary infections of prolonged CKD such as albuminuria and toxin production through the ML algorithm. These secondary infections also place a negative impact, especially on diabetics and patients with high blood pressure. Therefore, Determination of the role of ML algorithms in detecting CKD-associated diseases can be an effective research. Further research on effective treatment prediction and nutritional chart prediction of CKD patients through ML algorithm needs to be done in the future. Advanced technologies such as CNN, ML, random forest, and different classifiers can be used for these aspects to increase the recovery rate in CKD. By following this way, researchers and medical care professionals can enhance their service quality in accurate CKD diagnosis and treatment. Effective detection of CKD through ML algorithm is rapid and cost-effective, and due to this reason, the method can gain large popularity in the future.

CHAPTER 13

APPENDIX

SOURCE CODE

Chronic disease Prediction.ipynb

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import sklearn as sk
import seaborn as sns
```

# Reading CSV file from Local Drive

```
df=pd.read_csv(r"C:\Users\monis\Downloads\kidney_disease.csv");
```

#Describe Data

```
df.shape
df.head()
df.tail()
df.describe(include="all")
df.info()
df.dtypes
```

#Data Preprocessing

```
# isnull function
df.isnull().sum()
df.corr()
plt.figure(figsize=(15,8));
plt.title("Correlation",color="green")
sns.heatmap(df.corr(),linewidth=1,annot=True);
```

```
#Checking No of Null Values Through Visualization
import missingno as msn
msn.bar(df,color="red");
df.isnull()
# duplicated function
df.duplicated().value_counts()

# Finding Count Of CKD And Not CKD And Changing CKT/T Values To CKD
df['classification'].value_counts()
df['classification'].unique()
df[df["classification"]=="ckd\t"]
df["classification"]=df["classification"].replace("ckd\t","ckd",regex=True)
plt.figure(figsize=(17,7))
sns.countplot(data=df, x="classification")
plt.title("\nChronic Kidney Disease Distribution\n", fontsize=25)
plt.show();
df["age"].isnull().sum()
df["age"]=df["age"].fillna(df["age"].mean())
df.info()
numerical=[]
for col in df.columns:
    if df[col].dtype=="float64":
        numerical.append(col)
print(numerical)
for col in df.columns:
    if col in numerical:
        df[col].fillna(df[col].median(), inplace=True)
    else:
        df[col].fillna(df[col].mode()[0], inplace=True)
```

# Label Encoder

```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
object_col = [col for col in df.columns if df[col].dtype == 'object']
for col in object_col:
    df[col] = le.fit_transform(df[col])
df.info()
df.head()
df.columns
```

# **Data Visualization**

```
fig, ax = plt.subplots(figsize=(6,7))
M=df[['bu']]
N=df[['classification']]
plt.title("Relation Between Blood Urea And Chronic Kidney Disease",color="red");
plt.xlabel("Blood Urea",color="green")
plt.ylabel("Chronic Kidney Disease(1=Yes,0=No)",color="green")
ax.scatter(M,N);
plt.show();

fig, ax = plt.subplots(figsize=(6,7))
M=df[['sc']]
N=df[['classification']]
plt.title("Relation Between Serum Creatine And Chronic Kidney Disease");
plt.xlabel("Serum Creatine")
plt.ylabel("Chronic Kidney Disease(1=Yes,0=No)")
ax.scatter(M,N);
plt.show();

fig, ax = plt.subplots(figsize=(6,7))
M=df[['htn']]
N=df[['classification']]
plt.title("Relation Between Hypertension: yes And Chronic Kidney Disease");
plt.xlabel("Hypertension: yes")
plt.ylabel("Chronic Kidney Disease(1=Yes,0=No)")
ax.scatter(M,N);
plt.show();

fig, ax = plt.subplots(figsize=(6,7))
M=df[['al']]
N=df[['classification']]
plt.title("Relation Between Albumin And Chronic Kidney Disease");
plt.xlabel("Albumin")
plt.ylabel("Chronic Kidney Disease(1=Yes,0=No)")
ax.scatter(M,N);
plt.show();

fig, ax = plt.subplots(figsize=(6,7))
M=df[['dm']]
N=df[['classification']]
plt.title("Relation Between Diabetes Mellitus: yes And Chronic Kidney
Disease",color="red");
plt.xlabel("Diabetes Mellitus: yes")
plt.ylabel("Chronic Kidney Disease(1=Yes,0=No)")
ax.scatter(M,N);
plt.show();

sns.boxplot(x=df['classification'], y=df['bu'])
plt.show();

sns.scatterplot(data=df,x="su",y="htn",hue='classification');

sns.catplot(x="htn",y="su",data=df,kind="box");
plt.xlabel("Sugar",color="red")
plt.ylabel("Classification",color="red")
plt.title("Boxplot of Hypertension and Sugar",color="green");
plt.figure(figsize=(20,10))
sns.boxplot(data=df, x="ane", y="hemo", palette='seismic')
plt.xlabel("Hemeoglobin",color="red")
plt.ylabel("Aneamia",color="red")
plt.show()

plt.figure(figsize=(20,10))
sns.boxplot(data=df, y='hemo', x="pcc", hue="ane")
plt.xlabel("Hameoglobin",color="red")
plt.ylabel("Pus Cell Clumps",color="red")
plt.show()

df.columns
```

# Get Unique Values(Class or Labels) in y variable

```
df['classification'].shape

df.groupby("classification").mean()

df.columns
```

# Defining Target or Dependent Variable (y) and Feature or Independent Variables (X)

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

from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.25,random_state=222)
print("Training Data ::-")
print("The shape of X training data is :-" ,X_train.shape)
print("The shape of y training data is :-" ,y_train.shape)
print("Testing Data ::-")
print("The shape of X testing data is :-" ,X_test.shape)
print("The shape of y testing data is :-" ,y_test.shape)
```

#Checking the correlated variables using heatmap(Pearson Correlation)

```
import seaborn as sns
plt.figure(figsize=(20,10))
cor = X_train.corr()
sns.heatmap(cor, annot=True, cmap=plt.cm.CMRmap)
plt.show();

def correlation(dataset, threshold):
    col_corr = set()  # Set of all the names of correlated columns
    corr_matrix = dataset.corr()
    for i in range(len(corr_matrix.columns)):
        for j in range(i):
            if abs(corr_matrix.iloc[i, j]) > threshold: # we are interested in absolute coeff value
                colname = corr_matrix.columns[i]  # getting the name of column
                col_corr.add(colname)
    return col_corr

corr_features = correlation(X_train, 0.75)
len(set(corr_features))

corr_features

X_train.drop(corr_features,axis=1)
X_test.drop(corr_features,axis=1)
```

# Redefining the feature variables

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

# Standarlization of X variables

```
from sklearn.preprocessing import StandardScaler
sss=StandardScaler()
X=sss.fit_transform(X)
X.shape
```

# **Train Test Split**

```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.25,random_state=222)



print("Training Data ::-")
print("The shape of X training data is :-" ,X_train.shape)
print("The shape of y training data is :-" ,y_train.shape)

print("Testing Data ::-")
print("The shape of X testing data is :-" ,X_test.shape)
print("The shape of y testing data is :-" ,y_test.shape)
```

#Modeling

# 1. Logistic Regression

```
from sklearn.linear_model import LogisticRegression
model=LogisticRegression(max_iter=200,random_state=222)
model

model.fit(X_train,y_train)
```

# **Prediction**

```
y_predic=model.predict(X_test)
print(y_predic)

sns.countplot(y_predic);
```

# Probability of Each Predicted Class

```
model.predict_proba(X_test)
```

#Model Evaluation

```
from sklearn.metrics import confusion_matrix, classification_report,accuracy_score

print("Accuracy of the model is :  %3f " % accuracy_score(y_test,y_predic))

print(confusion_matrix(y_test,y_predic))

print(classification_report(y_test, y_predic))
```

# ***2.*** **Decision Tree Classifier Algorithm**

# **Modeling**

```
from sklearn.tree import DecisionTreeClassifier
model=DecisionTreeClassifier(random_state=222)

model.fit(X_train,y_train)
```

# **Prediction**

```
y_predict=model.predict(X_test)
print(y_predict)

print(model.predict_proba(X_test))
```

# **Model Evaluation**

```
from sklearn.metrics import confusion_matrix, classification_report,accuracy_score

print(confusion_matrix(y_test,y_predict))

print(classification_report(y_test, y_predict))
```

# Plotting the Decision Tree

```
from sklearn.tree import plot_tree
import matplotlib.pyplot as plt
plt.figure(figsize=(10,25))
plot_tree(model,filled=True);
```

# 3. K-Nearest Neighbors

# Modeling

```
from sklearn.neighbors import KNeighborsClassifier
model=KNeighborsClassifier()

model.fit(X_train,y_train)
```

#Prediction

```
y_predict=model.predict(X_test)
print(y_predict)
```

# Probability of Each Predicted Class

```
print(model.predict_proba(X_test))
```

# Model Evaluation

```
from sklearn.metrics import confusion_matrix, classification_report,accuracy_score

print(confusion_matrix(y_test,y_predict))

print(classification_report(y_test, y_predict))
```

Flask web app.ipynb

```
<html>
        <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
                <title>Flaskimio</title>
        <style>
        body {
        background: linear-gradient(
                        rgba(20,20,20, .75),
                        rgba(20,20,20, .75)),
                        url(
        'https://pharmanewsintel.com/images/site/article_headers/_normal/Chronic_Disease_
        Manage.png');
        .container {
          border: 2px solid #ccc;
          padding: 10px;
          width: 20em;
        height:21em;
        background-color:white;
        }
        .hello{
        opacity: 0.5;
        }
        </style>
        </head>
        <body>
        <marquee bgcolor="white">IBM NALAIYA THIRAN</marquee>
        <center><p style="font-size:60px;color:white;">Chronic Disease
        Prediction</p></center>
        <form action="/val" method="post"><center>
          <label for="age">Age:</label><br>
          <input type="number" id="age" name="age"><br><br><br>
```

```
 <label for="bp">Blood Pressure:</label><br>
 <input type="number" id="bp" name="bp">
<br>
<br>
<br>
 <label for="sg">Urinary Specific Gravity:</label><br>
 <input type="number" id="sg" name="sg">
<br>
<br>
<br>
 <label for="al">al:</label><br>
 <input type="number" id="al" name="al">
<br>
<br>
<br>
 <label for="su">su:</label><br>
 <input type="number" id="su" name="su">
<br>
<br>
<br>
 <label for="rbc">rbc:</label><br>
 <input type="text" id="rbc" name="rbc">
<br>
<br>
<br>
 <label for="pc">pc:</label><br>
 <input type="text" id="pc" name="pc">
<br>
<br>
<br>
 <label for="pcc">pcc:</label><br>
 <input type="text" id="pcc" name="pcc">
```

```
<br>
<br>
<br>
  <label for="ba">ba:</label><br>
  <input type="text" id="ba" name="ba">
<br>
<br>
<br>
  <label for="bgr">bgr:</label><br>
  <input type="number" id="bgr" name="bgr">
<br>
<br>
<br>
  <label for="bu">bu:</label><br>
  <input type="number" id="bu" name="bu">
<br>
<br>
<br>
  <label for="sc">sc:</label><br>
  <input type="number" id="sc" name="sc">
<br>
<br><br>
  <label for="sod">sod:</label><br>
  <input type="number" id="sod" name="sod">
<br>
<br>
<br>
  <label for="pot">pot:</label><br>
  <input type="number" id="pot" name="pot">
<br>
<br>
<br>
  <label for="hemo">hemo:</label><br>
  <input type="number" id="hemo" name="hemo">
```

```
<br>
<br>
<br>
  <label for="pcv">pcv:</label><br>
  <input type="text" id="pcv" name="pcv">
<br>
<br>
<br>
  <label for="wc">wc:</label><br>
  <input type="text" id="wc" name="wc">
<br>
<br>
<br>
  <label for="rc">rc:</label><br>
  <input type="text" id="rc" name="rc">
<br>
<br>
<br>
  <label for="htn">htn:</label><br>
  <input type="text" id="htn" name="htn">
<br>
<br>
<br>
  <label for="dm">dm:</label><br>
  <input type="text" id="dm" name="dm">
<br>
<br>
<br>
  <label for="cad">cad:</label><br>
  <input type="text" id="cad" name="cad">
<br>
<br>
<br>
```

```
            <label for="appet">appet:</label><br>
             <input type="text" id="appet" name="appet">
            <br>
            <br>
            <br>
             <label for="pe">pe:</label><br>
             <input type="text" id="pe" name="pe">
            <br>
            <br>
            <br>
             <label for="ane">ane:</label><br>
             <input type="text" id="ane" name="ane">
            <br>
            <br></center>
             <center><button type="submit">Check</button></center>
            </form>
            </body>
</html>
```

rename.html

```
<html>
            <head>
            <style>
            body {
             background-color: #E6E6FA;
            }
            </style>
            </head>
            <body >
            <br>
            <br>
            <br>
            <center><h1>{{answer1}}</h1></center>
            <br>
```

```
        <center><h1>{{answer2}}</h1></center>
        <center><img alt="Qries" src="https://image.shutterstock.com/image-vector/cute-
        illustration-sick-kidney-characters-260nw-1529381825.jpg"
              width=500" height="500"></center>
        </body>
</html>
```

rename2.html

```
<html>
        <head>
        <style>
        body {
          background-color: #E6E6FA;
        }
        </style>
        </head>
        <body >
        <br>
        <br>
        <br>
        <center><h1>{{answer1}}</h1></center>
        <br>
        <center><h1>{{answer2}}</h1></center>
        <center><img alt="Qries" src="https://media.istockphoto.com/vectors/vector-kidney-
        cartoon-human-body-health-organ-smiling-mascot-on-vector-
        id1136533246?k=20&m=1136533246&s=612x612&w=0&h=eWjhtoVOfX3lBF4fcs
        DD4ZLLzjeJ_Ax4cgdgdUexG1Q="
              width=500" height="500"></center>
        </body>
</html>
```

GitHub and Project Video Demo Link

GitHub link	**https://github.com/IBM-EPBL/IBM-Project-21249-659775830**
Project Video Demo Link	https://drive.google.com/file/d/1b7IJPOGS4Zym8--F1O74sHPC7aTJoIh0/view?usp=share_link