

**Early Prediction of Chronic Kidney Disease  
Disease Analysis Using Machine Learning**

**IBM PROJECT**

**Team Id:PNT2022TMID39435**

**TEAM LEAD**

**M.Aakash-510119205001**

**TEAM MEMBERS**

**V.DeepanRaj-510119205004**

**D.Prakash-510119205011**

**B.Rajesh-510119205013**

**OF**

**BACHELOR OF TECHNOLOGY**

**IN**

**INFORMATION TECHNOLOGY**

**ADHIPARASAKTHI COLLEGE OF  
ENGINEERING,KALAVAI-632506.**

## **Introduction to Chronic Kidney disease**

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.

The kidneys' main job is to filter extra water and wastes out of your blood to make urine. To keep your body working properly, the kidneys balance the salts and minerals—such as calcium, phosphorus, sodium, and potassium—that circulate in the blood. Your kidneys also make hormones that help control blood pressure, make red blood cells, and keep your bones strong.

Kidney disease often can get worse over time and may lead to kidney failure. If your kidneys fail, you will need dialysis or a kidney transplant to maintain your health.

The sooner you know you have kidney disease, the sooner you can make changes to protect your kidneys.

The kidneys' main job is to filter extra water and wastes out of your blood to make urine. To keep your body working properly, the kidneys balance the salts and minerals—such as calcium, phosphorus, sodium, and potassium—that circulate in the blood. Your kidneys also make hormones that help control blood pressure, make red blood cells, and keep your bones strong.

## **PROJECT OVERVIEW:**

The main goal of treatment is to prevent progression CKD to complete kidney failure. The best way to do this is to diagnose CKD early and control the underlying cause. The symptoms, evaluation, and management of CKD will be reviewed here. To predict the detection of disease at the earliest stage of spread of disease. To provide correct accuracy of spread of disease. To prevent spread of disease at the early stage.

## **PURPOSE:**

- To predict the disease spread in the early stage
- To prevent the loss of life and to prevent the kidney failure.
- To estimate the accuracy of chronic kidney disease.
- To save the time and to predict in an easier way.
- Your test results can be used to determine how damaged your kidneys are known as the stage of ckd.
- To prevent side effects of ckd such as breathing problems etc...

# **LITERATURE SURVEY OF CHRONIC KIDNEY DISEASE ANALYSIS:**

## **SURVEY 1:**

### **1. STATISTICAL AND DATA MINING ASPECTS ON KIDNEY STONES: A SYSTEMATIC REVIEW AND META ANALYSIS:**

This project is about a systematic review and meta analysis using classification algorithms studies predicted good accuracy with C4.5, classification tree and Random forest (93%) followed by Support Vector Machines (SVM) (91.98%). Logistic and NNge has also shown good accuracy results also shown good accuracy results with zero relative absolute error and 100% correctly classified results. Machine Learning approaches may provide better results in the treatment of kidney stones. Data mining offers a more quantitative approach to quality control with more uniform, user friendly for clinicians in reading the reports and reduce the errors. A meta analysis combines results of a number of studies that deal with a set of related research hypotheses. A meta analysis may be conducted on a several clinical trials of a medical treatment which refer to statistical methods combining evidence. In the present experimentation, we had analyzed a set of parameters related to kidney stone formation collected from patients in Kaviti, and Andhra Pradesh, India.

## **SURVEY 2:**

### **PREDICTION OF CHRONIC KIDNEY DISEASE USING RANDOM FOREST MACHINE LEARNING ALGORITHM:**

In this paper they have used random forest machine learning algorithm to predict the chronic kidney disease they have compared the performance of six classifiers in the prediction of chronic kidney disease analysis. The experimental results of the proposed method have demonstrated the RF has produced superior prediction performance in terms of classification accuracy. AUC and MCC respectively for our considered dataset. It was also observed that few classifiers have yielded poor classification accuracy as compared to RF like SMO and RBF.

## **SURVEY 3:**

### **A NOVEL DETECTION FOR KIDNEY DISEASE USING IMPROVED SUPPORT VECTOR MACHINE:**

This paper is about a novel detection for kidney disease using improved support vector machine. In this work, kidney disease prediction system was developed using classification algorithms (KNN, Naive Bayes, SVM, ISVM) through MATLAB data mining tool to predict effective and better

accurate results regarding whether the patient is suffering from kidney disease or not. As the kidney disease patients are increasing world-wide each year and huge amounts of data is available for research, where different data mining techniques are used in the diagnosis of kidney disease. Different attributes are used for prediction of kidney disease.

## **SURVEY 4:**

### **DATA MINING CLASSIFICATION ALGORITHMS FOR KIDNEY DISEASE PREDICTION:**

In this paper data mining classification algorithms for kidney disease prediction naive bayes, svm, ann, and fis they have used kidney function test (KFT) dataset. The algorithm which has the higher accuracy with the minimum execution time has been chosen as the best algorithm machine learning tool is resulting in high classification accuracy rate. The gap identified in the classifiers shows different accuracy rate. Data mining is an approach which dispense an intermixture of technique to identify a block of data or decision making knowledge in the database and eradicating these data in such a way that they can be put to use in decision support, forecasting and estimation.

## **REFERENCES**

- Kunwar V, Chandel K, Sai Sabitha, Bansal A(2016) Chronic Kidney Disease Analysis using Data Mining Classification Techniques. 2016 6th International Conference Cloud System and Big Data Engineering.
- Amirgaliyev Y, Shamiluulu S, Serek A(2018) Analysis of Chronic Kidney Disease Dataset by Applying Machine Learning Methods. 2018 IEEE 12th International Conference on Application of Information and Communication Technologies(AICT).
- Devika R, Sai Vaishnavi A, Subramaniya Swamy V(2019) Comparative Study of Classifier for Chronic Kidney Disease Prediction using Naive Bayes, KNN and Random Forest, 2019 3rd International Conference on Computing Methodologies and Communication(ICCMC).
- Alijaaf AJ, Al-jumeily D, Haglan HM, Alloghani M, Basker T, et al. (2018) Early Prediction of chronic kidney using machine learning supported by predictive analytics. 2018 IEEE Congress on Evolutionary Computation (CEC).

- Avci E, karakus S, Ozmen O, Avci(2018)Performance Comparison of Some Classifiers on Chronic Kidney Disease Data. 2018 6th International Symposium on Digital Forensic and Security (ISDFS).

### **Problem statement Definition:**

Our customer is a person who is suffering from breathing problems. He wishes to know whether there is spread of disease because breathing problem is a side effect or symptoms of kidney disease. He has too some test and he wishes to know the presence of disease using the given data. If there is a spread of disease, if yes he has to consult a doctor else he can feel free and satisfied. Due to the customer is very aged he cannot go to hospital and wants to detect in a simple manner.

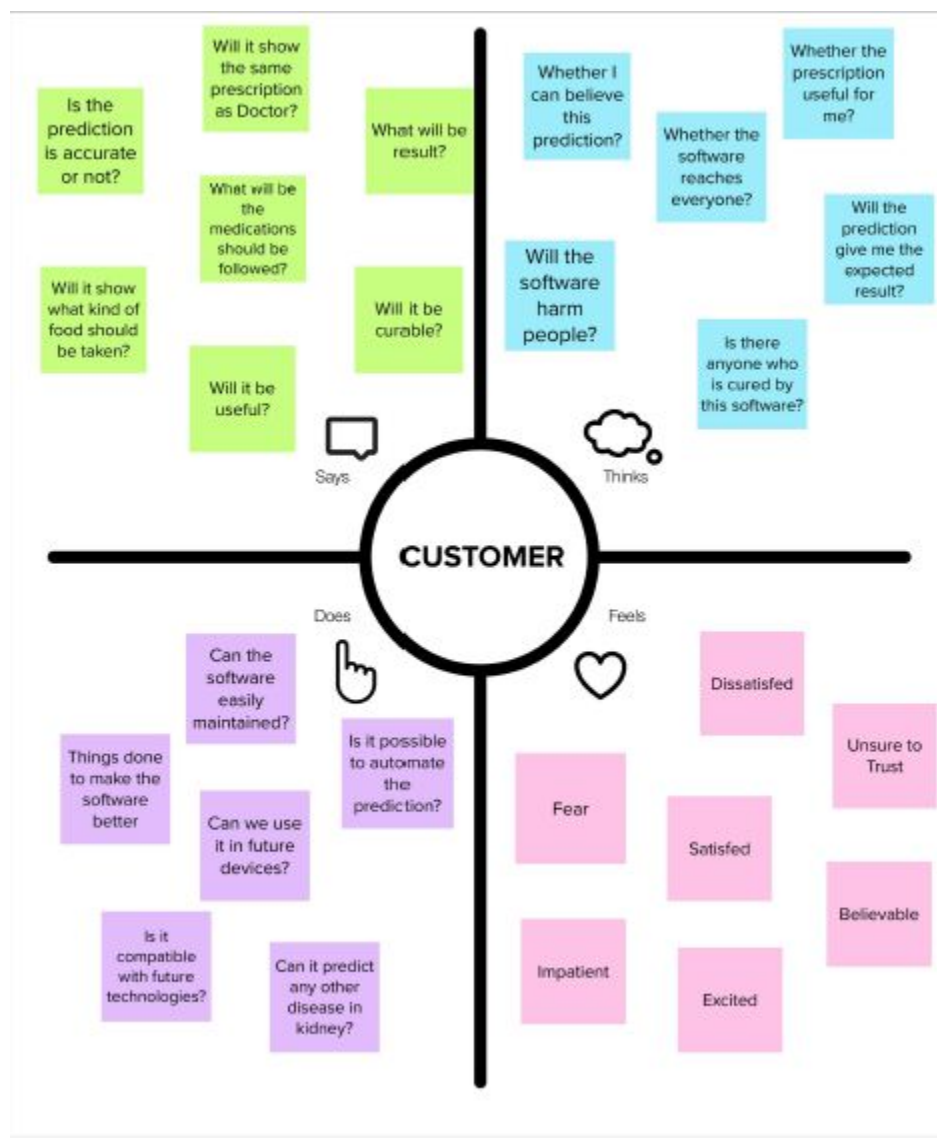
The prediction of disease can be done by using classification and regression methods.

### **IDEATION AND PROPOSED SOLUTION**



## Empathy Map Canvas:

An empathy map is a collaborative tool teams can use to gain a deeper insight into their customers. Much like a user persona, an empathy map can represent a group of users, such as a customer segment. The empathy map was originally created by Dave Gray and has gained much popularity within the agile community.



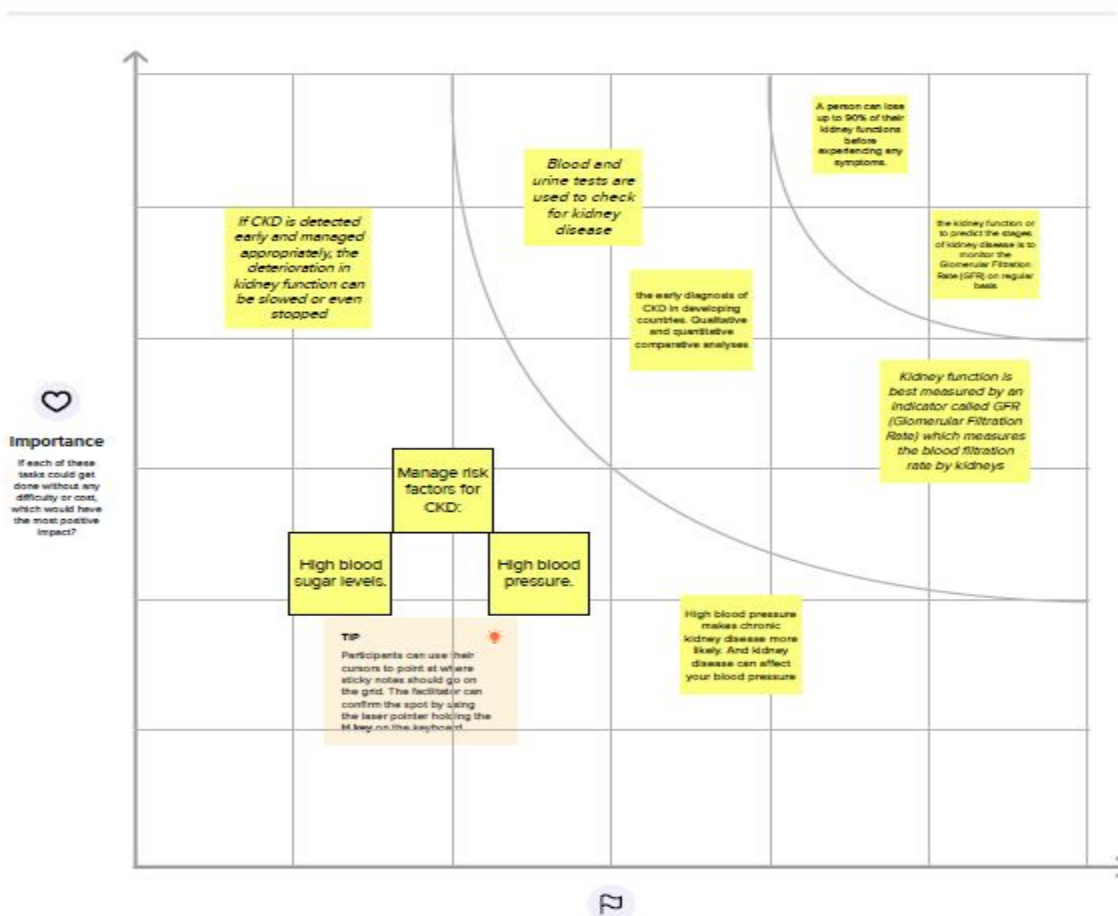
# Ideation and Brainstorming:

## Ideation:

Ideation is the process of forming ideas from conception to implementation, most often in a business setting. Ideation is expressed via graphical, written, or verbal methods, and arises from past or present knowledge, influences, opinions, experiences, and personal convictions.

## Brainstorming:

Brainstorming is a group activity where everyone comes together to discuss strategies for growth and improvement. You can exchange ideas, share important information and use these meetings as informal catch-up sessions with your co-workers.



## Proposed Solution:

Proposed Solution means the combination of software, hardware, other products or equipment, and any and all services (including any installation, implementation, training, maintenance and support services) necessary to implement the solution described by Vendor in its Proposal.

S.No	Parameter	Description
1.	Problem Statement (Problem to be solved)	The goal is to predict the presence of chronic kidney disease at the early stage of disease, so that the disease can be cured at the early stage and the prevention of loss of life can be done.
2.	Idea / Solution description	This concept is useful in medical field especially using this the disease can be predicted easily and quicker manner. The detection of this disease may help many patients to prevent additional side effects like pulmonary edema which may lead to breathing problems and heart attacks.

3.	Novelty / Uniqueness	Chances of kidney failure can be reduced and the disease can be cured. The side effects of the chronic kidney disease can be prevented by detecting at the early stage.
4.	Social Impact / Customer Satisfaction	It helps the doctors to predict the disease at the early stage and easier manner.  It helps to prevent loss of life and kidney failure.
5.	Scalability of the Solution	Supportful in prediction of disease and side effects of kidney disease.

## Problem Solution Fit:

The Problem-Solution Fit simply means that you have found a problem with your customer and that the solution you have realized for it actually solves the customer's problem.

Define CS, fit into C

Focus on JSP, fit into BE, understand RC

1. CUSTOMER SEGMENT(S)

CS

My customer is a 50 years old person who is suffering with kidney disease who needs to know the accuracy or spread of disease and to know the possibility of kidney failure.

6. CUSTOMER CONSTRAINTS

CC

The customer wants to predict the accuracy or presence of kidney disease because early treatment of disease may affect the health of customer and the treatment may be based on stage of disease either in early stage or in further stage. Hence the customer could not take treatment for chronic kidney disease.

5. AVAILABLE SOLUTIONS

AS

The customer may have dialysis treatment for chronic kidney disease and to maintain healthy diet. The customer should avoid smoking habits and drinking habits for further spread of chronic kidney disease. Controlling the blood pressure is another alternative way for stopping the spread of chronic kidney disease. These are some of the available solutions.

Explore AS, differe

2. JOBS-TO-BE-DONE / PROBLEMS

JB

The main job to be done is to take effective treatment after prediction of this disease. The problems is this disease has some side effects like breathing disease, high blood pressure etc.. This disease may lead to pulmonary edema which is the side effect of chronic kidney disease and kidney failure is the another problem occurs due to chronic kidney disease.

9. PROBLEM ROOT CAUSE

RC

The main cause of this problem is that the customer may have breathing problems and heart disease. The another factor is that the customer may have smoking and drinking habits which lead to the cause of chronic kidney disease. Other much less common conditions that can cause CKD include inflammation, infections, genetics, or long-standing blockage to the urinary system (such as enlarged prostate or kidney stones).

7. BEHAVIOUR

BE

The customer should take effective blood pressure and diabetes test to analyse the prediction of kidney disease. While if the customer has breathing problems the customer should check and consult the doctor if they have these problems. If the customer is fatigue immediately he or she should consult doctor and should take effective medication or take some rest. The customer should take regular treatment for this disease.

Focus on JSP, fit into BE, understand RC

3. TRIGGERS

TR

The customer should take necessary steps for further spread of disease. The customer should consult with patients suffered with this disease and should take alternate actions for this disease.

4. EMOTIONS: BEFORE / AFTER

EM

The customer may feel depressed after knowing the presence of this disease and happy if the accuracy level is less else may feel depressed and sad if the level is high. The customer may have fear of death due to high level spread of this disease.

10. YOUR SOLUTION

ST

The customer should mainly avoid smoking and drinking habits after knowing the presence of chronic kidney disease. The customer should follow healthy diet for healthy life. The customer should periodically check the blood pressure and diabetes level. The customer should regularly practice breathing exercises to avoid breathing problems. The customer should do periodical checking of blood pressure and diabetes level. The treatment should be maintained continuously. There are some of the solutions.

8. CHANNELS of BEHAVIOUR

CH

The customer may get easily tensed due to high blood pressure and the customer may have high level of diabetes. The customer may look fatigue or inactive. The customer may often suffer with kidney pain and suffer with poor nutritional condition. The customer may often look tired and may suffer with anaemia. The customer may suffer with breathing problems.

## REQUIREMENT ANALYSIS:

Requirements analysis, also called requirements engineering, is the process of determining user expectations for a new or modified product. In software engineering, it is sometimes referred to loosely by names such as requirements gathering or requirements capturing.

## TYPES OF REQUIREMENTS:

1. Functional Requirements.
2. Non-functional Requirements.

## Functional Requirements:

Functional requirements define what a product must do, what its features and functions are.

- Home page.
- Prediction page.
- Result page.
- Anaconda prompt.

## Non-Functional Requirements:

Non-functional requirements are global constraints on a software system are e.g., development costs, operational costs, performance, reliability, maintainability, portability, robustness etc.

Some of the non functional requirements are,

- Security
- Reliability
- Compatibility
- Environment friendly
- Maintainability
- Usability

# **PROJECT DESIGN:**

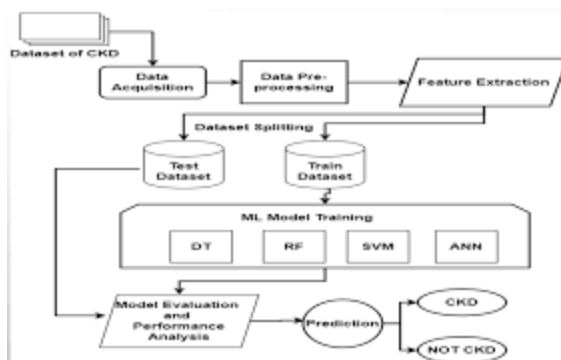
Project design is an early phase of the project lifecycle where ideas, processes, resources, and deliverables are planned out. A project design comes before a project plan as it's a broad overview whereas a project plan includes more detailed information.

## **TOPICS IN PROJECT DESIGN:**

- DATA FLOW DIAGRAMS.
- SOLUTION AND TECHNICAL ARCHITECTURE.
- USER STORIES.

## **DATA FLOW DIAGRAMS:**

A data flow diagram (DFD) maps out the flow of information for any process or system. It uses defined symbols like rectangles, circles and arrows, plus short text labels, to show data inputs, outputs, storage points and the routes between each destination.



## SOLUTION AND TECHNICAL ARCHITECTURE:

### SOLUTION ARCHITECTURE:

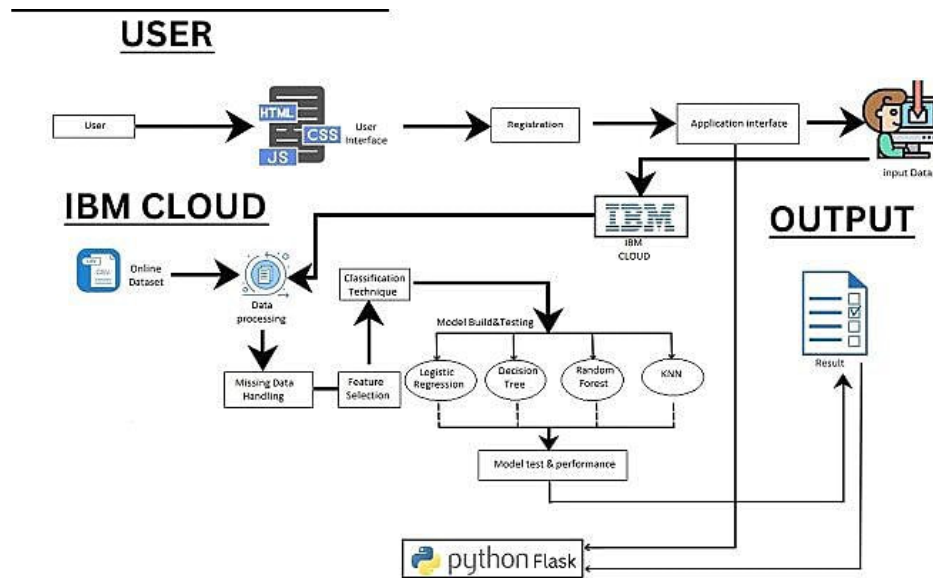
A solution architecture (SA) is an architectural description of a specific solution. SAs combine guidance from different enterprise architecture viewpoints (business, information and technical), as well as from the enterprise solution architecture.



### TECHNICAL ARCHITECTURE:

Technical architecture—which is also often referred to as application architecture, IT architecture, business architecture, etc, refers to creating a structured software solution that will meet the business needs and expectations while providing a strong technical plan for the growth of the software application through its lifetime.





## USER STORIES:

- A user story is an informal, general explanation of a software feature written from the perspective of the end user or customer. The purpose of a user story is to articulate how a piece of work will deliver a particular value back to the customer.
- Ex: As a customer, I want to know the spread of the kidney disease. By entering the values like red blood count, pulmonary edema values I can know the accuracy of spread of disease.
- As a user, I can identify the disease easier.
- As a user, I can predict from diabetes as per prescription.
- As a user, I can use it to clearly identify the kidney stones.

## **PROJECT PLANNING AND SCHEDULING:**

### **PROJECT PLANNING:**

Project planning is a discipline addressing how to complete a project in a certain timeframe, usually with defined stages and designated resources. One view of project planning divides the activity into these steps: setting measurable objectives. identifying deliverables. scheduling.

### **SCHEDULING:**

Scheduling is the process of arranging, controlling and optimizing work and workloads in a production process or manufacturing process. Scheduling is used to allocate plant and machinery resources, plan human resources, plan production processes and purchase materials.

### **SPRINT PLANNING:**

Sprint Planning is an event that defines what can be delivered in the upcoming Sprint and how their work can be achieved. It kicks off the Sprint. Each Sprint has a specific duration.

- Sprint 1-Data Collection.
- Sprint 2- Model Building.
- Sprint 3- Training and Testing.
- Sprint 4- Implementation of the Application.

### **ESTIMATION:**

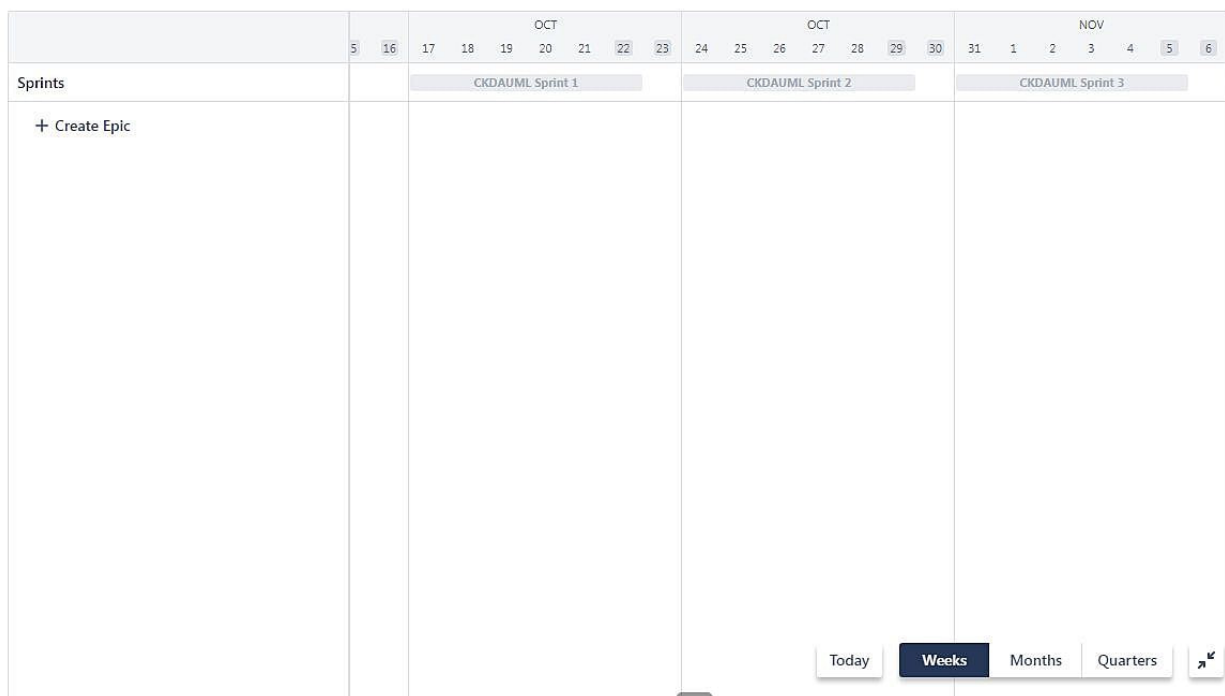
Estimation is a process to predict the time and the cost that a project requires to be finished appropriately.

## SPRINT DELIVERY SCHEDULE:

Sprint Delivery Schedule is the process of describing the duration of each sprint and the delivery of each sprint is called sprint delivery schedule.

Sprint	Sprint Start Date	Sprint End Date	Sprint Release Date
Sprint-1	24 Oct 2022	29 Oct 2022	29 Oct 2022
Sprint-2	31 Oct 2022	05 Nov 2022	05 Nov 2022
Sprint-3	07 Nov 2022	12 Nov 2022	12 Nov 2022
Sprint-4	14 Nov 2022	19 Nov 2022	19 Nov 2022

## Reports From Jira:



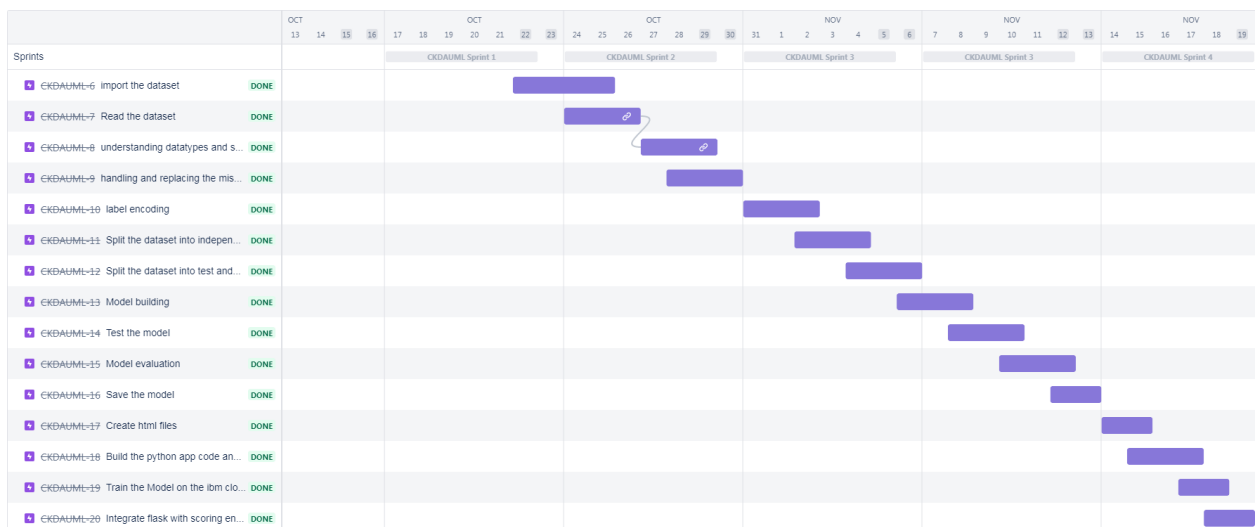
## Backlog

Search Epic Insights

CKDAUML Sprint 5 22 Nov – 26 Nov (4 issues)

0 0 0 Complete sprint

- CKDAUML-1 Logistic regression has an error DONE
  - CKDAUML-3 Html page is not running DONE
  - CKDAUML-4 Python code has error DONE
  - CKDAUML-2 Prediction is not accurate DONE
- [+ Create issue](#)



## CODING AND SOLUTIONING:

Coding or programming is the key activity and an engineering methodology through which the system visualized by the end user in terms of requirements is brought to life.

Solutioning is the process of acquiring the solution to the given problem is called as solutioning.

## FEATURE 1:

The main feature used in the coding is the machine learning concept. This machine learning is mainly used for prediction of accuracy of disease.

Machine learning is a subfield of artificial intelligence, which is broadly defined as the capability of a machine to imitate intelligent human behavior.

## FEATURE 2:

The second main feature used in coding is the Regression model.

Logistic Regression model is used for prediction of the chronic kidney disease.

Logistic regression aims to solve classification problems. It does this by predicting categorical outcomes, unlike linear regression that predicts a continuous outcome. In the simplest case there are two outcomes, which is called binomial, an example of which is predicting if a tumor is malignant or benign.

```
from sklearn.linear_model import LogisticRegression
lgr = LogisticRegression()
lgr.fit(x_train,y_train)
```

## **TESTING:**

Testing is the practice of making objective judgments regarding the extent to which the system (device) meets, exceeds or fails to meet stated objectives.

## TEST CASES:

A test case is a set of actions performed on a system to determine if it satisfies software requirements and functions correctly.

## TEST SCENARIOS:

- Verify Whether is able to enter data to prediction of accuracy of disease.
- Verify whether the user is getting the correct accuracy of disease.
- Verify whether the environment is user friendly or not.

## USER ACCEPTANCE TESTING:

User acceptance testing (UAT), also called application testing or end-user testing, is a phase of software development in which the software is tested in the real world by its intended audience.

It involves testing the entire software and detecting the errors, measuring the level of security, environment friendly software etc.

## **RESULTS:**

### Performance Metrics:

Performance metrics are defined as figures and data representative of an organization's actions, abilities, and overall quality.

### Label Encoding:

Label Encoding refers to converting the labels into a numeric form so as to convert them into the machine-readable form.

```
from sklearn.preprocessing import LabelEncoder # importing Labelencoding from sklearn
for i in catcols: # looping through all the categorical column
    print("LABEL ENCODING OF:",i)
    LEi = LabelEncoder() # creating an object of labelencoder
    print(c(data[i])) # getting the classes values before transformation
    data[i] = LEi.fit_transform(data[i]) # transferring our test classes to numerical values
    print(c(data[i])) # getting the classes values after transformation
    print("***100")
```

## Independent and Dependent Variables:

A dependent variable is a variable whose value depends on another variable, whereas An Independent variable is a variable whose value never depends on another variable.

```
selcols=['red_blood_cells','pus_cell','blood glucose random','blood_urea','pedal_edema','anemia','diabetesmellitus','coronary_artery_disease']
x=pd.DataFrame(data,columns=selcols)
y=pd.DataFrame(data,columns=['class'])
print(x.shape)
print(y.shape)
```

## Build The Model:

Model building is the process of developing a probabilistic model that best describes the relationship between the dependent and independent variables.

```
from sklearn.linear_model import LogisticRegression
lgr = LogisticRegression()
lgr.fit(x_train,y_train)
```

## Accuracy Score Of The Model:

It is the process of predicting the accuracy score of the model.

```
accuracy_score(y_test,y_pred)
```

```
0.8625
```

## Confusion Matrix of our Model:

A Confusion matrix is an  $N \times N$  matrix used for evaluating the performance of a classification model, where  $N$  is the number of target classes.

```
conf_mat = confusion_matrix(y_test,y_pred)
conf_mat
```

```
array([[40, 10],
       [ 1, 29]])
```

## **ADVANTAGES AND DISADVANTAGES:**

### **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.
- To prevent the kidney failure.
- To prevent loss of life.
- Using this we are able to predict at the early stage and the patient can take efficient treatment according to the spread.

### **DISADVANTAGES OF CHRONIC KIDNEY DISEASE:**

- Sometimes the accuracy may vary.
- The result may vary if the accuracy value is wrong or vary.

These are some of the advantages of disadvantages of chronic kidney disease analysis using machine learning.



## **CONCLUSION:**

Chronic renal failure represents a critical period in the evolution of chronic renal disease and is associated with complications and comorbidities that begin early in the course of the disease. These conditions are initially subclinical but progress relentlessly and may eventually become symptomatic and irreversible. Early in the course of chronic renal failure, these conditions are amenable to interventions with relatively simple treatments that have the potential to prevent adverse outcomes. Globally, CKD is most commonly attributed to diabetes and/or hypertension, but other causes such as glomerulonephritis, infection, and environmental exposures (such as air pollution, herbal remedies, and pesticides) are common in Asia, sub-Saharan Africa, and many developing countries. Genetic risk factors may also contribute to CKD risk. For example, sickle cell trait and the presence of 2 APOL1 risk alleles, both common in people of African ancestry but not European ancestry, may double the risk of CKD.<sup>4,7–10</sup>.

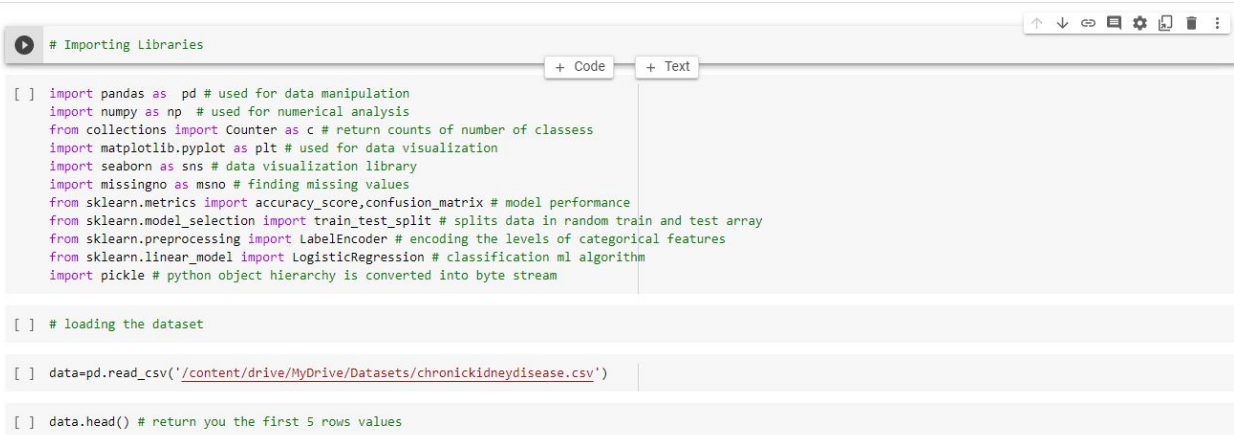
## **FUTURE SCOPE:**

- In future, it can be used in hospitals.
- It can be used at medical fields for easy prediction of kidney disease at the early stage.
- It can be used as a webservice for prediction using online.
- It can be used to develop an application which may be helpful for prediction of disease at the easiest way.
- For old age people it is very helpful to predict the disease.
- Easy way of prediction of kidney disease.

## APPENDIX:

A document that describes the design of a software component, product, or system.

### SOURCE CODE:



```
[ ] # Importing Libraries
import pandas as pd # used for data manipulation
import numpy as np # used for numerical analysis
from collections import Counter as c # return counts of number of classess
import matplotlib.pyplot as plt # used for data visualization
import seaborn as sns # data visualization library
import missingno as msno # finding missing values
from sklearn.metrics import accuracy_score, confusion_matrix # model performance
from sklearn.model_selection import train_test_split # splits data in random train and test array
from sklearn.preprocessing import LabelEncoder # encoding the levels of categorical features
from sklearn.linear_model import LogisticRegression # classification ml algorithm
import pickle # python object hierarchy is converted into byte stream

[ ] # loading the dataset

[ ] data=pd.read_csv('/content/drive/MyDrive/Datasets/chronickidneydisease.csv')

[ ] data.head() # return you the first 5 rows values
```

```
[ ] data.head() # return you the first 5 rows values
```

	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 x 26 columns

```
[ ] data.tail() # return you the last 5 rows values
```

	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

```
data.head(10) # return the first 10 rows values
```

	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	5	60.0	90.0	1.015	3.0	0.0	NaN	NaN	notpresent	notpresent	...	39	7800	4.4	yes	yes	no	good	yes	no	ckd
6	6	68.0	70.0	1.010	0.0	0.0	NaN	normal	notpresent	notpresent	...	36	NaN	NaN	no	no	no	good	no	no	ckd
7	7	24.0	NaN	1.015	2.0	4.0	normal	abnormal	notpresent	notpresent	...	44	6900	5	no	yes	no	good	yes	no	ckd
8	8	52.0	100.0	1.015	3.0	0.0	normal	abnormal	present	notpresent	...	33	9600	4.0	yes	yes	no	good	no	yes	ckd
9	9	53.0	90.0	1.020	2.0	0.0	abnormal	abnormal	present	notpresent	...	29	12100	3.7	yes	yes	no	poor	no	yes	ckd

10 rows x 26 columns

```
[ ] data.drop(["id"],axis=1,inplace=True) # drop is used for drop the column
```

```
[ ] data.columns # return all the column names
```

```
data.columns # return all the column names
```

```
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')
```

```
[ ] data.columns=['age','blood_pressure','specific_gravity','albumin','sugar','red_blood_cells','pus_cell','pus_cell_clumps','bacteria','blood_glucose_random',  
                 'blood_urea','serum_creatinine','sodium','potassium','hemoglobin','packed_cell_volume','white_blood_cell_count','red_blood_cell_count','hypertension',  
                 'diabetesmellitus','coronary_artery_disease','appetite','pedal_edema','anemia','class'] # manually giving the name of the columns  
data.columns
```

```
Index(['age', 'blood_pressure', 'specific_gravity', 'albumin', 'sugar',  
      'red_blood_cells', 'pus_cell', 'pus_cell_clumps', 'bacteria',  
      'blood glucose random', 'blood_urea', 'serum_creatinine', 'sodium',  
      'potassium', 'hemoglobin', 'packed_cell_volume',  
      'white_blood_cell_count', 'red_blood_cell_count', 'hypertension',  
      'diabetesmellitus', 'coronary_artery_disease', 'appetite',  
      'pedal_edema', 'anemia', 'class'],  
      dtype='object')
```

```
[ ] data.info() # info will give you a summary of dataset
```

```
# Target Column
```

```
[ ] data['class'].unique() # find the unique elements of an array
```

```
array(['ckd', 'ckd\t', 'notckd'], dtype=object)
```

```
[ ] # Rectifying the Target Column
```

```
[ ] data['class']=data['class'].replace("ckd\t","ckd") # replace is used for renaming  
data['class'].unique()
```

```
array(['ckd', 'notckd'], dtype=object)
```

```
[ ] catcols = set(data.dtypes[data.dtypes != 'O'].index.values) # only fetch the object type columns  
print(catcols)
```

```
{'hypertension', 'class', 'red_blood_cell_count', 'anemia', 'pus_cell_clumps', 'white_blood_cell_count', 'diabetesmellitus', 'pedal_edema', 'red_blood_cells', 'ba
```

```
[ ] for i in catcols:  
    print("Columns :",i)  
    print(c(data[i])) # using counter for checking the number of classes in the column  
    print('*'*120+'\n')
```

```
[ ] # Removing the Columns which are not Numerical
```

```
[ ] # Categorical Column
```

```
[ ] catcols.remove('red_blood_cell_count')  
catcols.remove('packed_cell_volume')  
catcols.remove('white_blood_cell_count')  
print(catcols)
```

```
{'hypertension', 'class', 'anemia', 'pus_cell_clumps', 'diabetesmellitus', 'pedal_edema', 'red_blood_cells', 'bacteria', 'coronary_artery_disease', 'pus_cell', 's
```

```
[ ] # Numerical Columns
```

```
[ ] contcols=set(data.dtypes[data.dtypes!='O'].index.values) # only fetch the float and int type columns  
contcols=pd.DataFrame(data,columns=contcols)  
print(contcols)
```

```
[ ] contcols.remove('specific_gravity')  
contcols.remove('albumin')  
contcols.remove('sugar')  
print(contcols)
```

```
{'blood_pressure', 'serum_creatinine', 'sodium', 'blood glucose random', 'blood_urea', 'age', 'hemoglobin', 'potassium'}
```

```
[ ] # Adding columns which we found continuous
```

```
[ ] contcols.add('red_blood_cell_count') # using add we can add the column  
contcols.add('packed_cell_volume')  
contcols.add('white_blood_cell_count')  
print(contcols)
```

```
{'blood_pressure', 'serum_creatinine', 'sodium', 'red_blood_cell_count', 'blood glucose random', 'white_blood_cell_count', 'blood_urea', 'age', 'hemoglobin', 'pac
```

```
[ ] # Adding columns which we found Categorical
```

```
[ ] catcols.add('specific_gravity')  
catcols.add('albumin')  
catcols.add('sugar')  
print(catcols)
```

```
[ ] # Rectifying the Categorical column classes
```

```
[ ] data['coronary_artery_disease'] = data.coronary_artery_disease.replace('\tno', 'no') # replacing \tno with no  
c(data['coronary_artery_disease'])  
  
Counter({'no': 364, 'yes': 34, nan: 2})
```

```
[ ] data['diabetesmellitus'] = data.diabetesmellitus.replace(to_replace={'\tno': 'no', '\tyes': 'yes', 'yes': 'yes'})  
c(data['diabetesmellitus'])  
  
Counter({'yes': 136, 'no': 261, 'yes': 1, nan: 2})
```

```
[ ] # Null Values
```

```
[ ] data.isnull().any() # it will return true if any missing values
```

```
[ ] data.isnull().sum() # returns the count of missing values
```

```
[ ] # Handling Continuous/numerical columns null values
```

```
data['blood_glucose_random'].fillna(data['blood_glucose_random'].mean(), inplace=True)  
data['blood_pressure'].fillna(data['blood_pressure'].mean(), inplace=True)  
data['blood_urea'].fillna(data['blood_urea'].mean(), inplace=True)  
data['hemoglobin'].fillna(data['hemoglobin'].mean(), inplace=True)  
data['packed_cell_volume'].fillna(data['packed_cell_volume'].mean(), inplace=True)  
data['potassium'].fillna(data['potassium'].mean(), inplace=True)  
data['red_blood_cell_count'].fillna(data['red_blood_cell_count'].mean(), inplace=True)  
data['serum_creatinine'].fillna(data['serum_creatinine'].mean(), inplace=True)  
data['sodium'].fillna(data['sodium'].mean(), inplace=True)  
data['white_blood_cell_count'].fillna(data['white_blood_cell_count'].mean(), inplace=True)
```

```
[ ] data['age'].fillna(data['age'].mode()[0], inplace=True)  
data['hypertension'].fillna(data['hypertension'].mode()[0], inplace=True)  
data['pus_cell_clumps'].fillna(data['pus_cell_clumps'].mode()[0], inplace=True)  
data['appetite'].fillna(data['appetite'].mode()[0], inplace=True)  
data['albumin'].fillna(data['albumin'].mode()[0], inplace=True)  
data['pus_cell'].fillna(data['pus_cell'].mode()[0], inplace=True)  
data['red_blood_cells'].fillna(data['red_blood_cells'].mode()[0], inplace=True)  
data['coronary_artery_disease'].fillna(data['coronary_artery_disease'].mode()[0], inplace=True)  
data['bacteria'].fillna(data['bacteria'].mode()[0], inplace=True)  
data['anemia'].fillna(data['anemia'].mode()[0], inplace=True)  
data['sugar'].fillna(data['sugar'].mode()[0], inplace=True)  
data['diabetesmellitus'].fillna(data['diabetesmellitus'].mode()[0], inplace=True)  
data['pedal_edema'].fillna(data['pedal_edema'].mode()[0], inplace=True)  
data['specific_gravity'].fillna(data['specific_gravity'].mode()[0], inplace=True)
```

```
[ ] data.isnull().sum()
```



```
[ ] # Label Encoding
```

```
[ ] from sklearn.preprocessing import LabelEncoder # importing Labelencoding from sklearn
for i in catcols: # looping through all the categorical column
    print("LABEL ENCODING OF:",i)
    LEi = LabelEncoder() # creating an object of labelencoder
    print(c(data[i])) # getting the classes values before transformation
    data[i] = LEi.fit_transform(data[i]) # transferring our test classes to numerical values
    print(c(data[i])) # getting the classes values after transformation
    print(""*100)
```

```
▶ # Creating Independent and Dependent
```

+ Code + Text

```
[ ] selcols=['red_blood_cells','pus_cell','blood_glucose_random','blood_urea','pedal_edema','anemia','diabetesmellitus','coronary_artery_disease']
x=pd.DataFrame(data,columns=selcols)
y=pd.DataFrame(data,columns=['class'])
print(x.shape)
print(y.shape)
```

```
(400, 8)
(400, 1)
```

```
[ ] # Splitting the Data into train and test
```

```
[ ] from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=None) # train test split the data
```

```
[ ] print(x_train.shape)
print(y_train.shape)
print(x_test.shape)
print(y_test.shape)
```

```
▶ # Build the Model
```

```
[ ] from sklearn.linear_model import LogisticRegression
lgr = LogisticRegression()
lgr.fit(x_train,y_train)
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/utils/validation.py:993: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please <
y = column_or_1d(y, warn=True)
/usr/local/lib/python3.7/dist-packages/sklearn/linear_model/_logistic.py:818: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
https://scikit-learn.org/stable/modules/linear\_model.html#logistic-regression
extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG,
LogisticRegression()
```

```
[ ] # Predicting our output with the model which we build
```

```
[ ] y_pred = lgr.predict(x_test)
```



```
# Accuracy score of Model
```

```
[ ] accuracy_score(y_test,y_pred)
```

```
0.8625
```

```
[ ] # Confusion Matrix of our Model
```

```
[ ] conf_mat = confusion_matrix(y_test,y_pred)
conf_mat
```

```
array([[40, 10],
       [ 1, 29]])
```

```
[ ] %cd /content/drive/MyDrive/Datasets
```

```
/content/drive/MyDrive/Datasets
```

```
[ ] #Dumping our model
pickle.dump(lgr, open('CKD.pk1','wb'))
```

Github Project Link:

<https://github.com/IBM-EPBL/IBM-Project-3678-1658588620>

Youtube Demo Video Link:

<https://youtu.be/Y6YqbC61x00>