Chronic Kidney Disease Prediction

Rohith J
HarshaVardhan R
Chendhuran N
Koushikravuru S

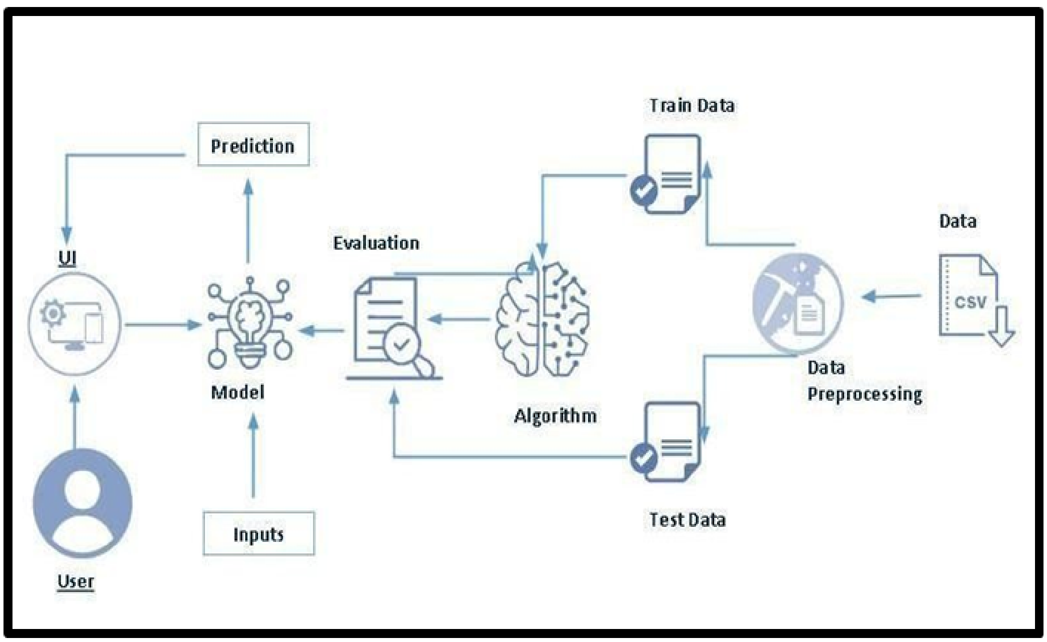
SmartInternz

www.smartinternz.com

Chronic Kidney Disease Detection Using Machine Learning

Chronic Kidney Disease (CKD) is a long-term condition characterized by gradual loss of kidney function. Early detection is crucial to prevent progression to kidney failure. However, CKD often remains undiagnosed due to subtle symptoms. This project aims to develop a machine learning model to predict CKD presence using clinical and laboratory data, assisting healthcare professionals in early diagnosis and treatment.

Technical Architecture:



Project Flow:

1. User interacts with the UI to enter the input.
2. Entered input is analysed by the model which is integrated.
3. Once model analysesthe input the prediction is showcased on the UI To accomplish this, we have to complete all the activities listed below,
4. Define Problem / Problem Understanding

   a. Specify the businessproblem
   b. Business requirements
   c. Literature Survey
   d. Social or Business Impact.
5. Data Collection &Preparation
   a. Collect the dataset
   b. Data Preparation
6. Exploratory Data Analysis
   a. Descriptive statistical
   b. Visual Analysis
7. Model Building
   a. Training the model in multiplealgorithms
   b. Testing the model
8. Performance Testing & Hyperparameter Tuning
   a. Testing model with multiple evaluation metrics
   b. Comparing model accuracybefore & after applying hyperparameter tuning
9. Model Deployment
   a. Save the best model
   b. Integrate with Web Framework
10. Project Demonstration &Documentation
    a. Record explanation Videofor project end to end solution
    b. Project Documentation-Step by step projectdevelopment procedure

Prior Knowledge:

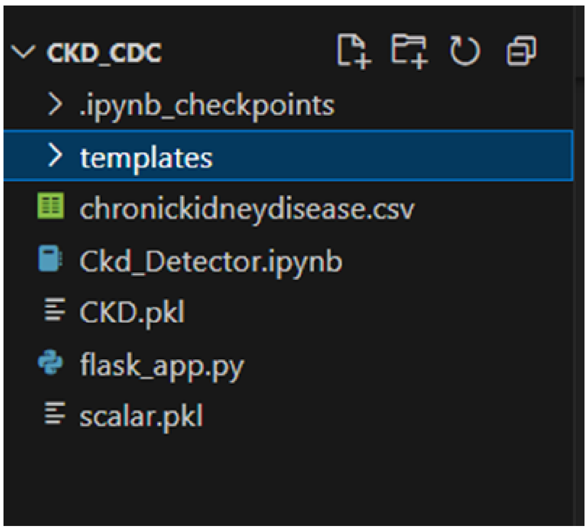
You must have prior knowledge of following topics to completethis project.

1. ML Concepts
   a. Supervised learning: https://www.javatpoint.com/supervised-machine-learning
   b. Unsupervised learning: https://www.javatpoint.com/unsupervised-machine-learning
2. Decision tree: https://www.javatpoint.com/machine-learning-decision-tree-classification-algorithm
3. Random forest: https://www.javatpoint.com/machine-learning-random-forest-algorithm
4. KNN: https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning
5. Xgboost: https://www.analyticsvidhya.com/blog/2018/09/an-end-to-end-guide-to-understand-the-math-behind-xgboost/

6. Evaluation metrics: https://www.analyticsvidhya.com/blog/2019/08/11-important-model-evaluation-error-metrics/
1. Flask Basics : https://www.youtube.com/watch?v=lj4I_CvBnt0

Project Structure:

Create the Project folderwhich contains files as shownbelow



Create the project folder with the following files and folders:

1. **data/:** Contains the CKD dataset CSV file
2. **templates/:** HTML files for the web interface (index.html, submit.html)
3. **app.py:** Flask backend script
4. **Ckd_Detector.ipynb:** Jupyter notebook with data analysis and model building
5. **model.pkl:** Serialized best model for deployment.

Milestone 1: DefineProblem / ProblemUnderstanding

Activity 1: Specify the business problem

CKD is a progressive condition that can lead to kidney failure if undetected early. The goal is to build a machine learning model that predicts CKD presence based on patient clinical data, enabling timely intervention.

Activity 2: Business requirements

A chronic kidney disease (CKD) detection project can have a variety of business requirements, depending on the specific goals and objectives of the project. Some potential requirements may include:

a. Accurate and up-to-date predictions: The project should use the most recent and reliable clinical and laboratory data to predict CKD, ensuring that the results are relevant and reflect current medical standards and practices.

b. Flexibility: The detection system should be flexible and able to adapt to new clinical indicators or diagnostic criteria as medical knowledge and guidelines evolve.

c. Compliance: The project should comply with all relevant healthcare regulations and data privacy laws, such as HIPAA or local health data protection standards, to ensure patient confidentiality and legal operation.

d. User-friendly interface: The CKD detection system should be easy to use and understand for both healthcare professionals and, where appropriate, patients, supporting informed clinical decisions and patient engagement..

Activity 3: Literature Survey

Recent literature demonstrates that machine learning (ML) techniques such as Decision Trees, Random Forests, Support Vector Machines (SVM), K-Nearest Neighbors (KNN), Logistic Regression, and ensemble methods have been widely applied for CKD detection and prognosis. Studies consistently report high accuracy rates—often exceeding 97%—when using these models on standard CKD datasets. For example, ensemble methods like Random Forest and Gradient Boosting have achieved accuracy rates above 99%, while deep learning models and hybrid approaches also show promise for further improving prediction performance.

A key focus in the literature is the importance of robust data preprocessing, including handling missing values, feature selection, and balancing datasets, as these steps significantly affect model performance. Feature engineering—such as the use of clinical indicators like hemoglobin, albumin, serum creatinine, and blood pressure—has been shown to enhance predictive power. In addition, interpretability remains a central concern: while complex models can offer higher accuracy, clinicians require explanations of model decisions to support trust and adoption in healthcare settings.

Despite these advances, several gaps and challenges persist. Many studies note the limitations of small sample sizes, lack of stage-specific predictions, and insufficient attention to model transparency and patient data privacy. There is also a need for more multi-class models that can predict specific CKD stages, as well as external validation on diverse populations to ensure generalizability.

In summary, the literature indicates that machine learning offers significant potential for improving early CKD detection and risk stratification. However, ongoing research is needed to address challenges related to data quality, model interpretability, and clinical integration, ensuring that such systems are both accurate and usable in real-world healthcare environments

Activity 4: Social or Business Impact.

Social Impact:
Improved patient care: By providing accurate and timely predictions of chronic kidney disease, a CKD detection project can help healthcare professionals make more informed decisions about diagnosis and treatment options. This leads to earlier interventions, better disease management, and improved patient outcomes.

Business Model/Impact:
Optimized healthcare resources: By identifying CKD at an early stage, the project can help healthcare providers allocate resources more efficiently, reduce the costs associated with late-stage treatments, and lower the overall burden on the healthcare system. Additionally, early detection can support preventive health programs and reduce hospital admissions related to advanced kidney disease.

Milestone 2: Data Collection &Preparation

ML dependsheavily on data.It is the most crucialaspect that makesalgorithm training possible. So, this section allows you to download the required dataset.

Activity 1: Collect the dataset

In this project, we have used a .csv dataset provided by our mentors. The dataset contains clinical records of patients relevant to Chronic Kidney Disease (CKD) detection. It includes features like blood pressure, specific gravity, albumin, sugar, red blood cells, etc.

Since the dataset was not downloaded from a public source like Kaggle or the

UCI repository, no external link is required. The data was pre-collected and shared as part of the academic mini-project.

As the dataset was readily available, we proceeded to read and analyze it using various visualization and statistical techniques to understand feature distributions, correlations, and patterns.

Note: While we have used key exploratory methods in this project, additional analysis techniques can always be incorporated for deeper insights.

Activity 1.1: Importing the libraries

Import the necessary librariesas shown in the image.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import pickle
import missingno as msno
from collections import Counter as c
from sklearn.metrics  import accuracy_score,confusion_matrix
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.linear_model import LogisticRegression
```

Activity 1.2: Read the Dataset

Our datasetformat might be in .csv, excel files, .txt, .json, etc. We can read the datasetwith the help of pandas.

In pandaswe have a function calledread_csv() to read the dataset.As a parameter we have to give the directory of the csv file.

1. For checking the null values, df.isna().any( ) function is used. To sum those null values we use .sum() function. From the belowimage we foundthat there are no null values present in our dataset. So we can skip handling the missing values step.

```
[2]: data=pd.read_csv('chronickidneydisease.csv')
```

```
[3]: data.head()
```

[3]:

	id	age	bp	sg	al	su	rbc	pc	pcc	ba	...	pcv	wc
0	0	48.0	80.0	1.020	1.0	0.0	NaN	normal	notpresent	notpresent	...	44	780(
1	1	7.0	50.0	1.020	4.0	0.0	NaN	normal	notpresent	notpresent	...	38	600(
2	2	62.0	80.0	1.010	2.0	3.0	normal	normal	notpresent	notpresent	...	31	750(
3	3	48.0	70.0	1.005	4.0	0.0	normal	abnormal	present	notpresent	...	32	670(
4	4	51.0	80.0	1.010	2.0	0.0	normal	normal	notpresent	notpresent	...	35	730(

Activity 2: Data Preparation

As we have understood how the data is, let's pre-process the collected data.

The downloaddata set is not suitablefor training the machine learningmodel as it might have so much randomness so we need to clean the dataset properly in order to fetch good results. This activity includes the following steps.

a. Handling missing values
b. Handling Outliers

Note: These are the general stepsof pre-processing the data beforeusing it for machine learning. Depending on the condition of your dataset, you may or may not have to go through all these steps.

Activity 2.1: Handling missingvalues

i. For checking the null values,df.isna().any( ) functionis used. To sum thosenull values we use .sum() function. From the below image we found that there are no null values present in our dataset. So we can skip handling the missing values step

[37]:
```
data.isnull().any()
```

[37]:
```
age                        True
blood_pressure             True
specific_gravity           True
albumin                    True
sugar                      True
red_blood_cells            True
pus_cells                  True
pus_cell_clumps            True
bacteria                   True
blood_glucose_random       True
blood_urea                 True
serum_creatinine           True
sodium                     True
potassium                  True
hemoglobin                 True
packed_cell_volume         True
white_blood_cell_count     True
red_blood_cell_count       True
hypertension               True
diabetesmillitus           True
coronory_artery_disease    True
appetite                   True
pedal_edema                True
anemia                     True
class                     False
```

Milestone 3: Exploratory Data Analysis

Activity 1: Descriptive statistical

Descriptive analysisis to study the basic features of data with the statistical process. Here pandas has a worthy function called describe. With this describe function we can understand the unique, top and frequent values of categorical features. And we can find mean, std, min, max and percentile values of continuous features.

[62]:
```
data.describe()
```

[62]:

	age	blood_pressure	specific_gravity	albumin	sugar	red_blood_cells	pus_cells	pus_cell_clumps	bacteria	blood_glucose_random
count	400.000000	400.000000	400.000000	400.00000	400.000000	400.000000	400.000000	400.000000	400.000000	400.000000
mean	51.675000	76.469072	1.017712	0.90000	0.395000	0.882500	0.810000	0.105000	0.055000	148.036517
std	17.022008	13.476298	0.005434	1.31313	1.040038	0.322418	0.392792	0.306937	0.228266	74.782634
min	2.000000	50.000000	1.005000	0.00000	0.000000	0.000000	0.000000	0.000000	0.000000	22.000000
25%	42.000000	70.000000	1.015000	0.00000	0.000000	1.000000	1.000000	0.000000	0.000000	101.000000
50%	55.000000	78.234536	1.020000	0.00000	0.000000	1.000000	1.000000	0.000000	0.000000	126.000000
75%	64.000000	80.000000	1.020000	2.00000	0.000000	1.000000	1.000000	0.000000	0.000000	150.000000
max	90.000000	180.000000	1.025000	5.00000	5.000000	1.000000	1.000000	1.000000	1.000000	490.000000

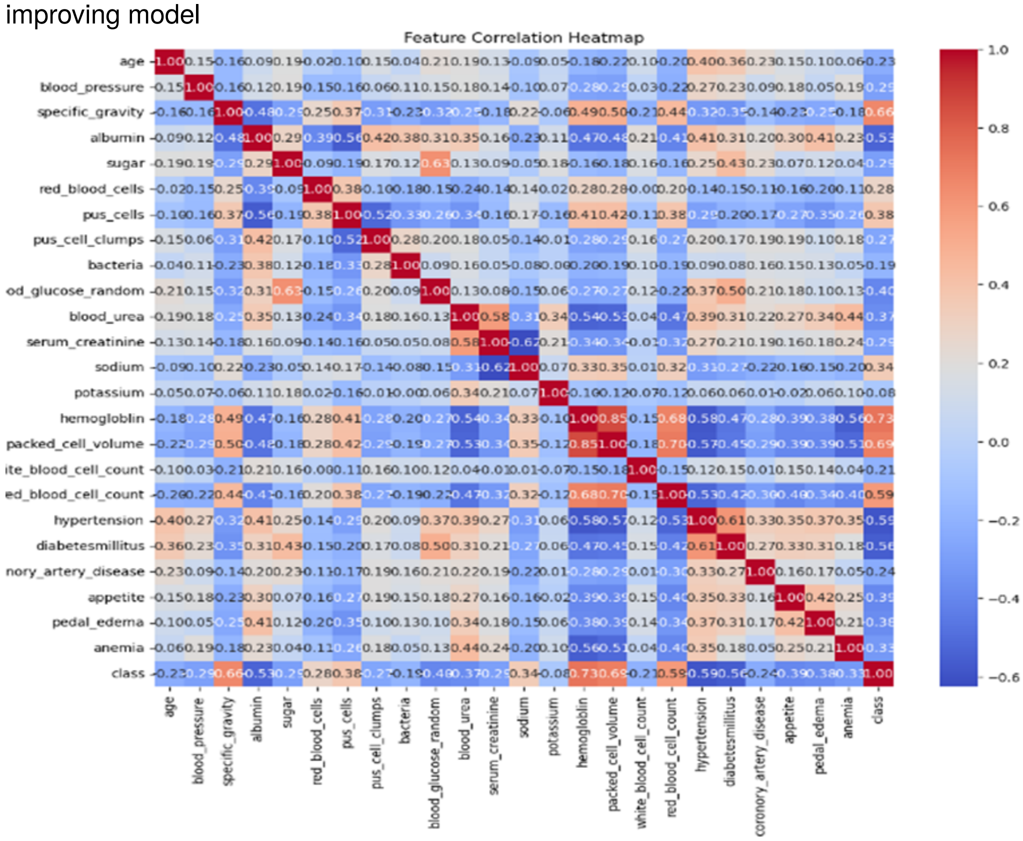
8 rows × 25 columns

Activity 2: Visual analysis

Visual analysisis the process of usingvisual representations, such as charts,plots, and graphs, to explore and understand data.It is a way to quickly identifypatterns, trends, and outliersin the data, which can help to gain insightsand make informed decisions.

Activity 2.1: Multivariate analysis

i. A correlation heatmap was generated to visualize the degree of linear relationships between variables. Key insights included: Strong negative correlation between hemoglobin levels and CKD status, indicating lower hemoglobin is associated with CKD. Features such as serum creatinine, blood urea, and albumin showed high correlation with the target variable, reinforcing their importance in kidney disease prediction. Multivariate analysis helped in feature selection by identifying redundant or highly correlated features, improving model



efficiency and accuracy.

Encoding the Categorical Features:

1. The categorical Features are can'tbe passed directlyto the Machine Learning Model.So we convert them into Numerical data based on their order. This Technique is called Encoding.
2. Here we are importingLabel Encoder from the SklearnLibrary.
3. Here we are applyingfit_transform to transformthe categorical featuresto numerical features.

```
[49]: for i in columns:
          print("Label Encoding of :",i)
          Lei=LabelEncoder()
          print(c(data[i]))
          data[i]=Lei.fit_transform(data[i])
          print(c(data[i]))
          print("*"*100)
```

Splitting data into trainand test

Now let'ssplit the Datasetinto train and test sets.First split the dataset into x and y and then splitthe data set

Here x and y variables are created. On x variable, df is passedwith dropping the target variable.And on y target variable is passed. For splitting training and testing data we are using train_test_split() function from sklearn. As parameters, we are passing x, y, test_size, random_state.

```
[54]: x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=2)
      print(x_train.shape)
      print(x_test.shape)
      print(y_train.shape)
      print(y_test.shape)

      (320, 8)
      (80, 8)
      (320, 1)
      (80, 1)
```

Scaling

i. Scaling is a technique used to transform the values of a dataset to a similar scale to improve the performance of machine learningalgorithms. Scaling is important becausemany machine learning algorithms are sensitive to the scale of the input features.

ii. Here we are using Standard Scaler.

iii. This scalesthe data to have a mean of 0 and a standarddeviation of 1. The formulais given by: X_scaled = (X - X_mean) / X_std

```
[56]: from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
x_train_scaled = scaler.fit_transform(x_train)
x_test_scaled = scaler.transform(x_test)
```

Milestone 4: ModelBuilding

Activity 1: Training the model

Since our model is designed for classification, we have chosen to use the logistic regression algorithm for our dataset.

Activity 1.1: Logistic Regression model

Logistic regression is a suitable method when the target variable is binary. Like other types of regression, it is used for making predictions. It helps in modeling the relationship between a binary dependent variable and one or more independent variables, which can be nominal, ordinal, interval, or ratio in nature. To train our logistic regression model, we utilize the x_train and y_train datasets obtained earlier from the train_test_split function. We apply the fit method to train the model using these inputs, as demonstrated below..

```
from sklearn.linear_model import LogisticRegression
from sklearn.linear_model import LogisticRegressionCV
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score, classification_report
lg = LogisticRegressionCV(solver='lbfgs', max_iter=5000, cv=10)
lg.fit(X_train, y_train)
print(confusion_matrix(y_test,lrg_pred))
```

Activity 2: Testing the model

Here we have tested with Decision Tree algorithm. You can test with all algorithm. With the help of predict() function.

```
[57]: y_pred=lgr.predict(x_test_scaled)
      print(y_pred)
      c(y_pred)

      [0 0 0 0 1 0 0 0 1 0 0 0 0 1 0 0 0 1 1 0 1 1 0 1 0 1 0 0 1 0 0 1 0 0 0 0 1
       0 0 1 0 1 0 0 0 1 0 1 1 0 0 0 0 1 0 0 0 1 1 0 0 1 1 0 0 0 0 1 1 1 1 0 0 0
       0 0 0 0 1 0]
[57]: Counter({0: 53, 1: 27})
```

Milestone 5: Performance Testing

Activity 1: Testing model with multipleevaluation metrics

the Logistic Regression model demonstrated the best performance, achieving an accuracy of 98.75% on the test dataset. The confusion matrix confirmed high true positive and true negative rates, and the classification report reflected strong precision, recall, and F1-scores for both CKD and non-CKD classes..

We analyzed performance using:

Confusion Matrix – to evaluate the model's ability to classify CKD vs non-CKD accurately.

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

[58]: array([[53,  1],
             [ 0, 26]], dtype=int64)
```

Classification Report – provided precision, recall, and F1-score for both

```
[63]: from sklearn.metrics import classification_report
      print(classification_report(y_test, y_pred))

                    precision    recall  f1-score   support

                 0       1.00      0.98      0.99        54
                 1       0.96      1.00      0.98        26

          accuracy                           0.99        80
         macro avg       0.98      0.99      0.99        80
      weighted avg       0.99      0.99      0.99        80
```

classes

.

Although hyperparameter tuning (e.g., using GridSearchCV or RandomizedSearchCV) was not applied in this version of the notebook, the selected model performed exceptionally well, suggesting robust default parameters. This step can be revisited in future iterations to further enhance performance.

Milestone 6: Model Deployment

Activity 1: Save the best model

Saving the best model aftercomparing its performance using different evaluation metrics means selecting the model with the highest performance.This can be useful in avoiding the need to retrain the model every time it is needed and also to be able to use it in the future.

```
[60]: pickle.dump(lgr,open('CKD.pkl','wb'))
```

```
[61]: pickle.dump(scaler,open('scalar.pkl','wb'))
```

Activity 2: Integrate with Web Framework

In this section, we will be building a web application that is integrated to the model we built. A UI is providedfor the uses where he has to enter the values for predictions. The enter values are given to the saved model and prediction is showcased on the UI.

This sectionhas the following tasks

1. Building HTML Pages
2. Building server-side script
3. Run the web application

Activity 2.1: Building Html Page:

For this project create HTML file namely

1. index.html

and save them in the templates folder. Refer this linkfor templates.

Activity 2.2: BuildPython code:

Import the libraries

Load the saved model.Importing the flaskmodule in the project is mandatory. An object of Flask class is our WSGI application. Flask constructor takes the name of the current module (name) as argument.

```
import pandas as pd
from flask import Flask, request, render_template
import pickle
```

```
app = Flask(__name__)

# Load model and scaler
model = pickle.load(open('CKD.pkl', 'rb'))
try:
    scaler = pickle.load(open('scalar.pkl', 'rb'))
except:
    scaler = None
```

Render HTML page:

```
@app.route('/')
def welcome():
    return render_template('index.html')
```

Here we will be using a declared constructor to route to the HTML page whichwe have created earlier.

In the above example, '/' URL is bound with the index.html function. Hence, when the home page of the web server is opened in the browser,the html page will be rendered. Whenever you enter the values from the html page the values can be retrieved using POST Method.

Retrieves the value from UI:

```
@app.route('/predict', methods=['POST'])
def predict():
    # Get form data
    features = [
        float(request.form['hemogloblin']),
        float(request.form['packed_cell_volume']),
        float(request.form['specific_gravity']),
        float(request.form['red_blood_cell_count']),
        float(request.form['hypertension']),
        float(request.form['diabetesmillitus']),
        float(request.form['albumin']),
        float(request.form['blood_glucose_random'])
    ]

    # Create DataFrame
    df = pd.DataFrame([features], columns=[
        'hemogloblin', 'packed_cell_volume', 'specific_gravity', 'red_blood_cell_
        'hypertension', 'diabetesmillitus', 'albumin', 'blood_glucose_random'
    ])

    # Scale and predict
    if scaler:
        df = scaler.transform(df)

    prediction = model.predict(df)[0]
    result = 'CKD Detected' if prediction == 0 else 'No CKD Detected'

    return render_template('result.html', prediction_text=result)
```

Here we are routingour app to predict() function. This function retrieves all the valuesfrom the HTML page using Post request. That is stored in an array. This array is passed to the model.predict() function. This function returns the prediction. And this prediction value will be rendered to the text that we have mentioned in the submit.html page earlier.

Main Function:

```
if __name__ == '__main__':
    app.run(debug=True)
```

Activity 2.3: Run the web application

a. Open anaconda promptfrom the start menu

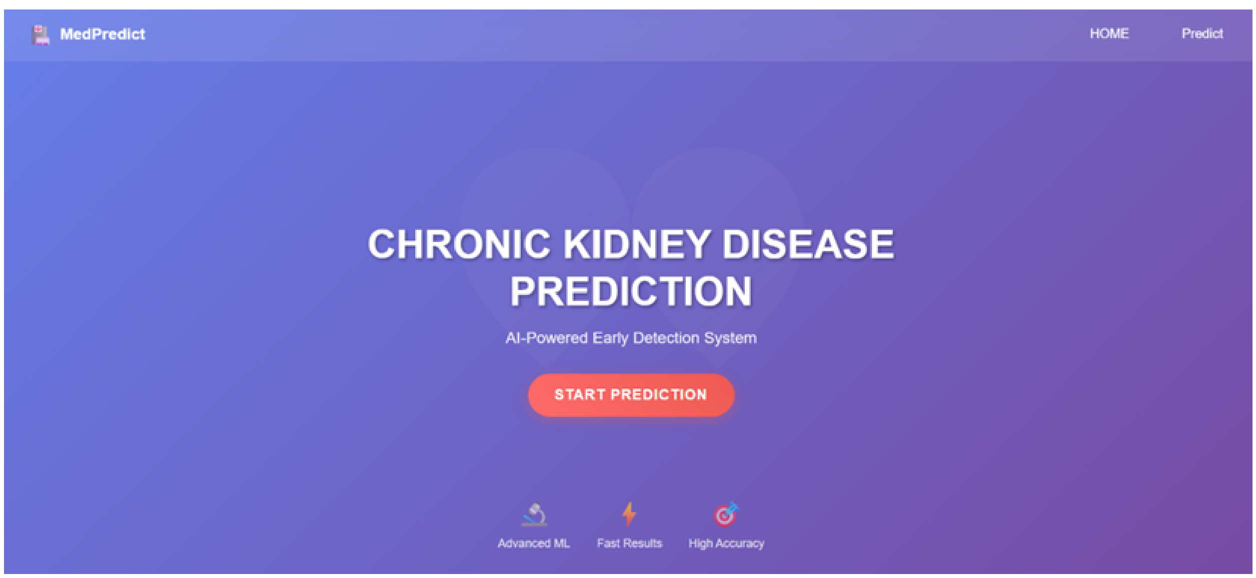
b. Navigate to the folder where your python script is.

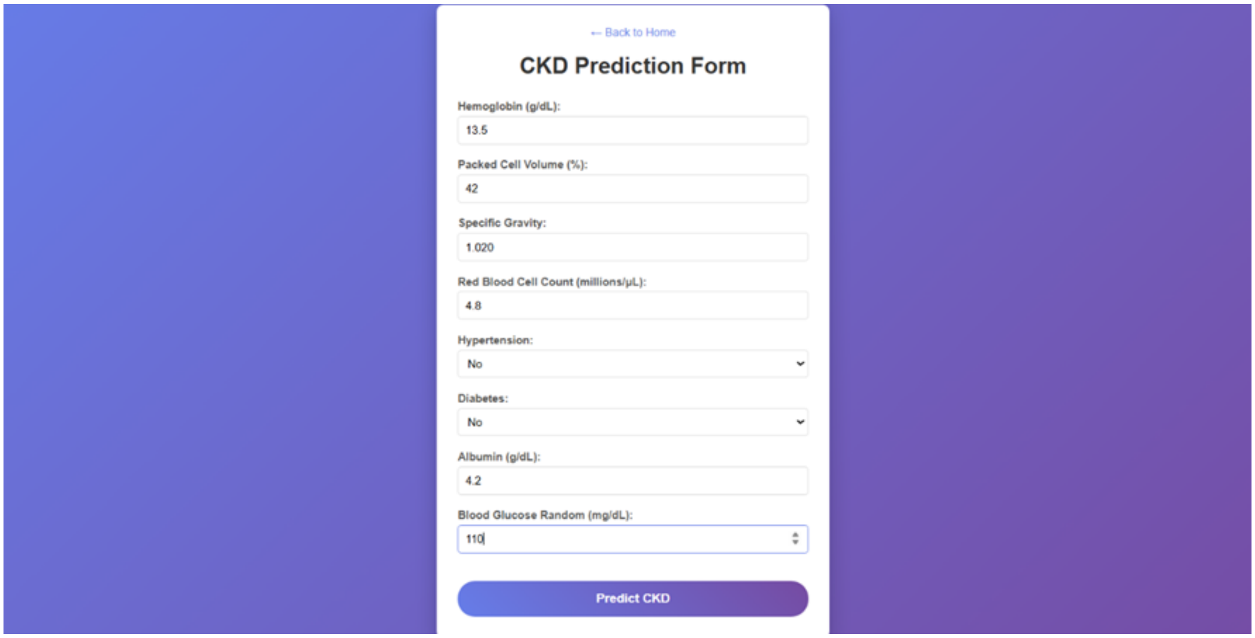
c. Now type “pythonapp.py” command

d. Navigate to the localhost where you can view your web page.

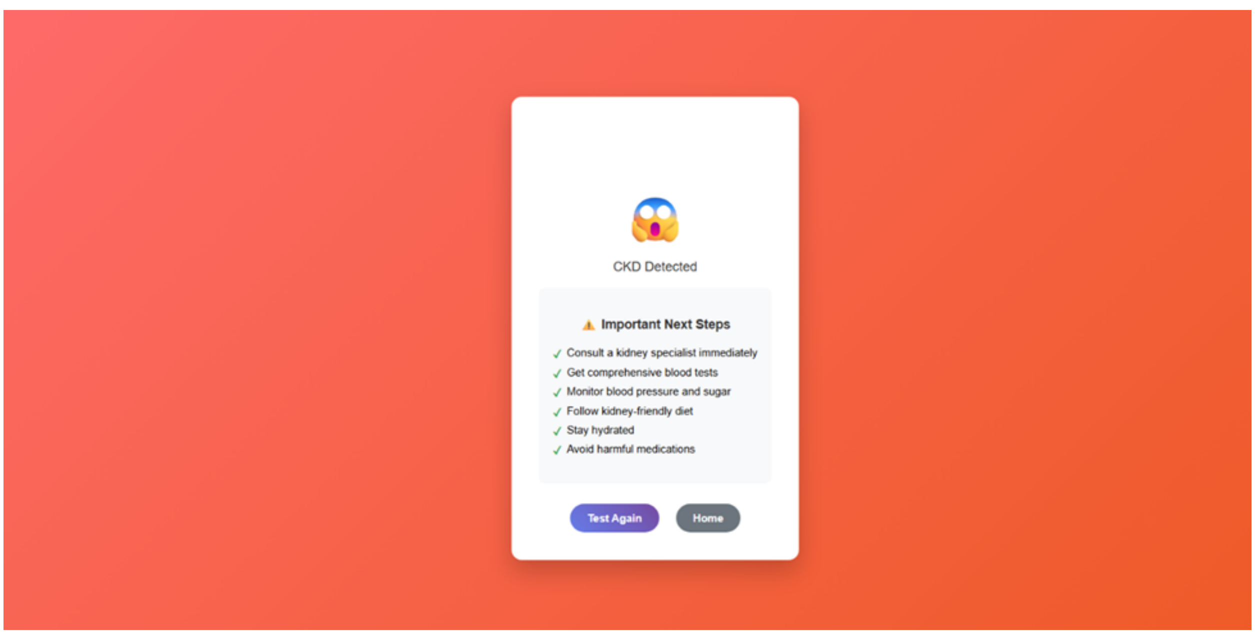
e. Click on the predict button from the top left corner, enter the inputs,click on the submit button, and see the result/prediction on the web.

Now,Go the web browserand write the localhost url (http://127.0.0.1:5000) to get the below result





← Back to Home
CKD Prediction Form
Hemoglobin (g/dL):
13.5
Packed Cell Volume (%):
42
Specific Gravity:
1.020
Red Blood Cell Count (millions/μL):
4.8
Hypertension:
No
Diabetes:
No
Albumin (g/dL):
4.2
Blood Glucose Random (mg/dL):
110
Predict CKD



CKD Detected
⚠️ Important Next Steps
✓ Consult a kidney specialist immediately
✓ Get comprehensive blood tests
✓ Monitor blood pressure and sugar
✓ Follow kidney-friendly diet
✓ Stay hydrated
✓ Avoid harmful medications
Test Again
Home