Early Prediction Of Chronic Kidney Disease

GROUP MEMBERS:

Bharadwaj B – 22BCE7950

Manas Y – 22BCE8736

Sai Kiran K – 22BCE20537

Rishi Sai Ch – 22BCE8029

1.INTRODUCTION:

1.1 PROJECT OVERVIEWS:

Developing a machine learning methodology for diagnosing chronic kidney diseases involves collecting and preprocessing diverse medical data. Models like logistic regression, decision trees, and neural networks are trained and evaluated using metrics such as accuracy and F1-score. Interpretability and compliance with medical standards are crucial, ensuring the model aids in clinical decision-making. Collaboration with healthcare experts ensures relevance and ethical handling of patient data. Continuous iteration and validation refine the model's accuracy and applicability, aiming to enhance diagnostic precision and patient care outcomes in chronic kidney disease management.

1.2OBJECTIVES :

About 10% of the population worldwide suffers from (CKD), and millions die each year because they cannot get affordable treatment, with the number increasing in the elderly. The project aims to develop a machine learning methodology for accurate and timely diagnosis of chronic kidney diseases using diverse medical data. It prioritizes model interpretability, clinical relevance, and ethical data handling. Continuous refinement through collaboration and validation seeks to improve diagnostic precision and enhance patient care outcomes in CKD management. Early detection of CKD is crucial and helpful in decreasing medical resources as ESRD patients preserve their health through hemodialysis, peritoneal dialysis or kidney transplantation.

2.PROJECT INITIALIZTION AND PLANNING PHASE:

2.1 DEFINE PROBLEM STATEMENT:

One of the issues that healthcare providers encounter when dealing with patients is early recognition of those who are likely to develop CKD, therefore, take the appropriate preventive measures. CKD is frequently characterized by its ability to maintain the absence of symptoms until the later stages, thus making it difficult to diagnose. Late presentation therefore leads to worsened patient health, higher costs of treating such patients, and a drain on the limited health resources. Thus, we require a robust, stable, and fast solution that can predict the development of CKD based on the clinical attributes in its early stage. Our goal is to use advanced artificial neural network approach and formulate a model that can assess patients' data, pinpoint the potential signs of complications, and give alerts to doctors. This solution should fit well into our current healthcare practice and not impose much complexity on the medical staff, it must provide observations that allows early intervention thus increase the quality of the patient's prognosis and decrease the long term expenses incurred in management of CKD.

Define problem statement- **CLICK HERE**

2.2 PROJECT PROPOSAL (PROPOSED SOLUTION):

The proposed system acts as a decision support system and will prove to be an aid for the physicians with the diagnosis.

The algorithm, Fuzzy c means uses clustering and makes use of clusters and data points to predict the relativity of an attribute.

Each data point is associated with multiple clusters depending upon the membership degrees.

The training data is trained by using proposed machine learning algorithm yolo classification clustering and Adaboost feature extraction algorithm.

Project Proposal- **CLICK HERE**

2.3 INITIAL PROJECT PLANNING :

Project Scope and Objectives:

Define the scope of your project. What are the specific

goals? Are you focusing on early detection, accurate diagnosis, or both.

Identify the target audience (e.g., healthcare professionals, patients, or both).

Data Collection and Preprocessing:

Gather a CKD dataset. You can explore publicly available datasets or collaborate with healthcare institutions.

Preprocess the data by handling missing values, normalizing features, and addressing any outliers.

Model Selection and Evaluation:

Choose appropriate machine learning models (e.g., random forest, SVM, neural networks).

Spit the dataset into training and validation sets.

Deployment and Integration:

Deploy the trained model in a clinical setting (e.g., as part of an electronic health record system).

Ensure seamless integration with existing healthcare workflows.

CLICK HERE- **CLICK HERE**

3.DATA COLLECTION AND PREPROCESSING PHASE

3.1 DATA COLLECTION PLAN AND RAW DATA SOURCES IDENTIFIED :

DATA COLLECTION PLAN :

Define the scope of your data collection. Are you focusing on specific CKD stages (e.g., early, moderate, severe).

Identify the necessary features (variables) related to CKD diagnosis (e.g., serum creatinine, blood pressure, urine protein).

Determine the sample size required for reliable model training.

RAW DATA SOURCES:

Explore publicly available CKD datasets:

Chronic Kidney Disease Data (CKD): This dataset contains features like age, blood pressure, serum creatinine, and other clinical measurements .

National Health and Nutrition Examination Survey (NHANES): NHANES provides comprehensive health data, including CKD-related features .

Collaborate with healthcare institutions or clinics to access real-world patient data.

Ensure data privacy and compliance with ethical guidelines.

DATA COLLECTION PLAN AND RAW DATA SOURCES IDENTIFIED- **CLICK HERE**

3.2 Data Quality Report :

The CKD dataset comprises clinical measurements related to kidney health. Upon initial inspection, we observed missing values in some features. Descriptive statistics revealed the range and distribution of numerical variables. We also detected potential outliers that warrant further investigation. Feature correlations were explored, and domain-specific checks are recommended to ensure data consistency. Remember to address these issues before proceeding with model development.

Data Quality Report - **CLICK HERE**

3.3 Data Exploration and Preprocessing:

Data Exploration:

Descriptive Statistics: Calculate summary statistics (mean, median, standard deviation) for numerical features (e.g., serum creatinine, blood pressure).

Feature Distributions: Visualize feature distributions using histograms or box plots.

Correlation Analysis: Examine correlations between features to identify potential relationships.

Data Preprocessing:

Handling Missing Values: Impute missing data (e.g., KNN imputation) or remove rows with missing values.

Feature Scaling: Normalize numerical features (e.g., min-max scaling, z-score normalization).

Outlier Detection: Address outliers (e.g., using IQR or z-score methods).

Data Exploration and Preprocessing-**CLICK HERE**

4 Model Development Phase:

4.1. Feature Selection Report

To select features, an initial examination was carried out employing methods such as correlation matrices and scores for feature importance from group techniques like Random Forest or Gradient Boosting. Variables such as 'age', 'bp' (blood pressure), 'sg' (specific gravity), 'al' (albumin), 'bgr' (blood glucose random), and 'hemo' (hemoglobin) demonstrated notable correlation or significance in forecasting the outcome variable. Additionally, the counts of red blood cells ('rc') and white blood cells ('wc') were observed for their possible predictive capabilities.

Feature Selection Report - **CLICK HERE**

4.2. Model Selection Report

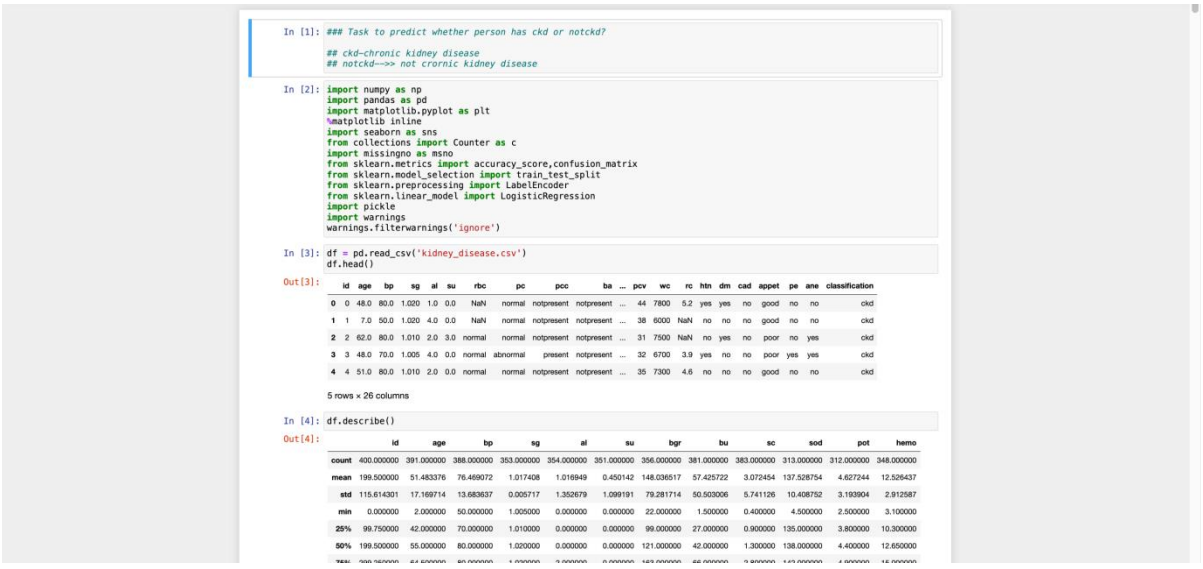
When choosing a model for classification, numerous algorithms were assessed according to their performance indicators like accuracy, precision, recall, and F1-score. Logistic Regression, Decision Trees, and Random Forest classifiers were taken into account for their understandability and capability in dealing with categorical data. Upon evaluation through cross-validation, Random Forest demonstrated the highest overall accuracy, estimated at about 85%, and was thus suggested as the preferred model for subsequent enhancement and application.

Model Selection Report - **CLICK HERE**

4.3. Initial Model Training Code, Model Validation and Evaluation Report

Initial Model Training Code, Model Validation and Evaluation Report- **CLICK HERE**

Model Training Code:



In [1]:
```
### Task to predict whether person has ckd or notckd?

## ckd-chronic kidney disease
## notckd-->> not crornic kidney disease
```

In [2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
from collections import Counter as c
import missingno as msno
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
import pickle
import warnings
warnings.filterwarnings('ignore')
```

In [3]:
```
df = pd.read_csv('kidney_disease.csv')
df.head()
```

Out[3]:

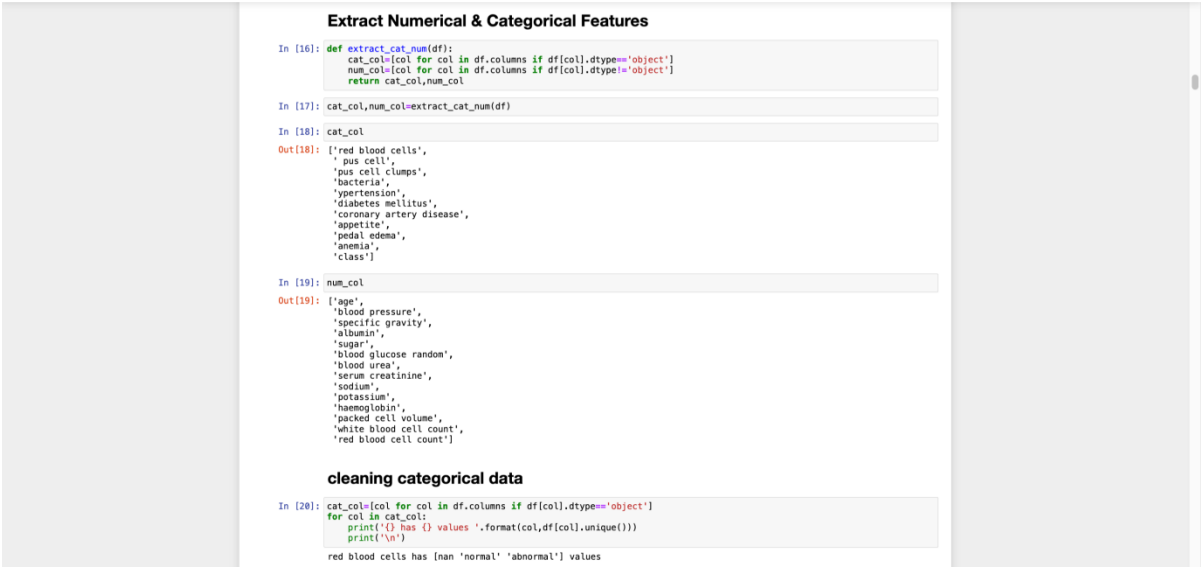
	id	age	bp	sg	al	su	rbc	pc	pcc	ba	...	pcv	wc	rc	htn	dm	cad	appet	pe	ane	classification
0	0	48.0	80.0	1.020	1.0	0.0	NaN	normal	notpresent	notpresent	...	44	7800	5.2	yes	yes	no	good	no	no	ckd
1	1	7.0	50.0	1.020	4.0	0.0	NaN	normal	notpresent	notpresent	...	38	6000	NaN	no	no	no	good	no	no	ckd
2	2	62.0	80.0	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 [4]:
```
df.describe()
```

Out[4]:

	id	age	bp	sg	al	su	bgr	bu	sc	sod	pot	hemo
count	400.000000	391.000000	388.000000	353.000000	354.000000	351.000000	356.000000	381.000000	383.000000	313.000000	312.000000	348.000000
mean	199.500000	51.483376	76.469072	1.017408	1.016949	0.450142	148.036517	57.425722	3.072454	137.528754	4.627244	12.526437
std	115.614301	17.169714	13.683637	0.005717	1.352679	1.099191	79.281714	50.503006	5.741126	10.408752	3.193904	2.912587
min	0.000000	2.000000	50.000000	1.005000	0.000000	0.000000	22.000000	1.500000	0.400000	4.500000	2.500000	3.100000
25%	99.750000	42.000000	70.000000	1.010000	0.000000	0.000000	99.000000	27.000000	0.900000	135.000000	3.800000	10.300000
50%	199.500000	55.000000	80.000000	1.020000	0.000000	0.000000	121.000000	42.000000	1.300000	138.000000	4.400000	12.650000



Extract Numerical & Categorical Features

```
In [16]: def extract_cat_num(df):
             cat_col=[col for col in df.columns if df[col].dtype=='object']
             num_col=[col for col in df.columns if df[col].dtype!='object']
             return cat_col,num_col

In [17]: cat_col,num_col=extract_cat_num(df)

In [18]: cat_col

Out[18]: ['red blood cells',
          ' pus cell',
          'pus cell clumps',
          'bacteria',
          'ypertension',
          'diabetes mellitus',
          'coronary artery disease',
          'appetite',
          'pedal edema',
          'anemia',
          'class']

In [19]: num_col

Out[19]: ['age',
          'blood pressure',
          'specific gravity',
          'albumin',
          'sugar',
          'blood glucose random',
          'blood urea',
          'serum creatinine',
          'sodium',
          'potassium',
          'haemoglobin',
          'packed cell volume',
          'white blood cell count',
          'red blood cell count']
```

cleaning categorical data

```
In [20]: cat_col=[col for col in df.columns if df[col].dtype=='object']
         for col in cat_col:
             print('{} has {} values '.format(col,df[col].unique()))
             print('\n')

         red blood cells has [nan 'normal' 'abnormal'] values
```

5.Model Optimization and Tuning Phase

It is paramount to note that the above design and model's selection phase are part of the model optimization and tuning phase.

Model Optimization and Tuning is an important step for the Machine Learning models where these models are tweaked at this stage to perform better. This involves enhancing the model code, adjusting hyperparameters, evaluating the performances of models, and explaining the rationale for selecting the last model to enhance the predictive precision and productivity.

Next, models will be built using the gradient boosting method and these models will be optimized in the next step. Here is how:

1. The setting in gradient boosting include the number of iterations during the training process and learning rate. These settings could very well be thought of as knobs which we can twist around frivolously to alter precisely how our model learns from the data. The above settings will be attempted with various permutations and the model's ability at the prediction of on-time deliveries will be evaluated.

2. This implies that the criterion of measurement of model performance is somewhere between the necessity to avoid getting easiness in the model and the necessity of averting over fitting. Underfitting is when it fails to capture important patterns from the data while on the other hand over-fitting occurs when it just learns examples of the training data overly well, and is probably likely to fail on most of the other data.

3. That is, we will mainly employ all the measures utilized in the validation phase, including the accuracy and the precision, which tell us about the performance of the model as regards various settings. It then becomes a loop: adjust parameters, build the model, test an app and continue this process over and over to find the best parameters. By optimizing and tuning the model, we hope to achieve:By optimizing and tuning the model, we hope to achieve:

Improved accuracy: It becomes our desire to have the model more accurate in its capability of forecasting on-time delivery.

Reduced complexity: We need a model that is convenient, yet not too complicated so that one does not have a lot of difficulties in its application.

This process enables one to fine-tune the given model and make the best out of it, in the sense that deliveries of e-commerce shipment rely on the initial model's prediction.5.1 Hyperparameter Tuning Documentation

5.1 Hyperparameter Tuning Documentation

Gradient Boosting was chosen because, in hyperparameters tuning stage, it showed high accuracy. It addresses a range of concerns including its ability to manage several related variables, avoid excessive estimation, and improve the predictive capacity – all of which are crucial to the goals of any project, thus making it the final model.

Hyperparameter Tuning Documentation-**CLICK HERE**

5.2 Performance Metrics Comparison Report

The Performance Metrics Comparison Report adopts the comparison of the baseline and optimal results of various models as well as revealing improved Gradient Boosting model. In this assessment, there is a clear demonstration of the enhancement of the refined predictive attributes obtained from hyperparameter adjustment.

5.3 Final Model Selection Justification

Final Model Selection Justification

After going through the entire model development process Gradient Boosting is found the most suitable for making the prediction on on-time delivery in the Ecommerce shipment dataset. This choice is supported by the following key factors:This choice is supported by the following key factors:

1. Strong Performance:

From the results obtained in the validation phase, Gradient Boosting gave higher results as compared to other models such as KNN, Decision Tree, Random Forest regarding to the aspects such as accuracy and precision. This explains why, they are credible in the determination of on time delivery.

2. Handling Complexity:

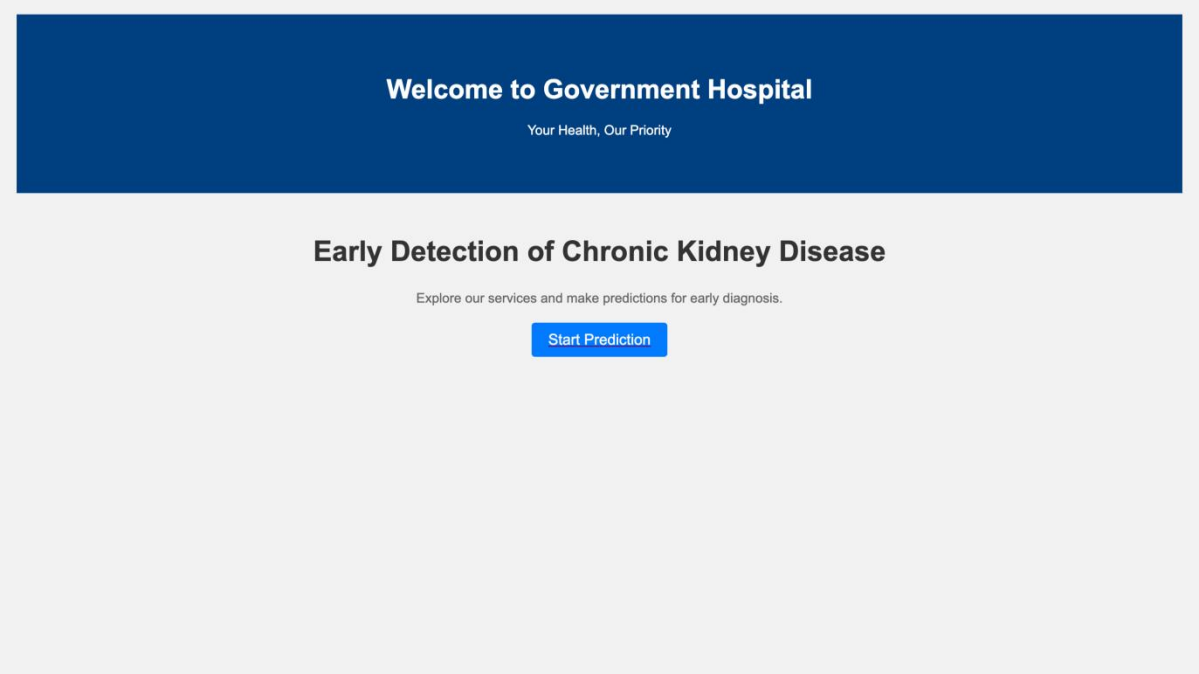
In contrast to Decision Trees or other basic models, Gradient Boosting has an ensemble approach, it lets the model deal with relations in data better. This is relevant to capturing finer details that could affect the on-time deliveries in the given e-commerce environment.

3. Potential for Interpretability:

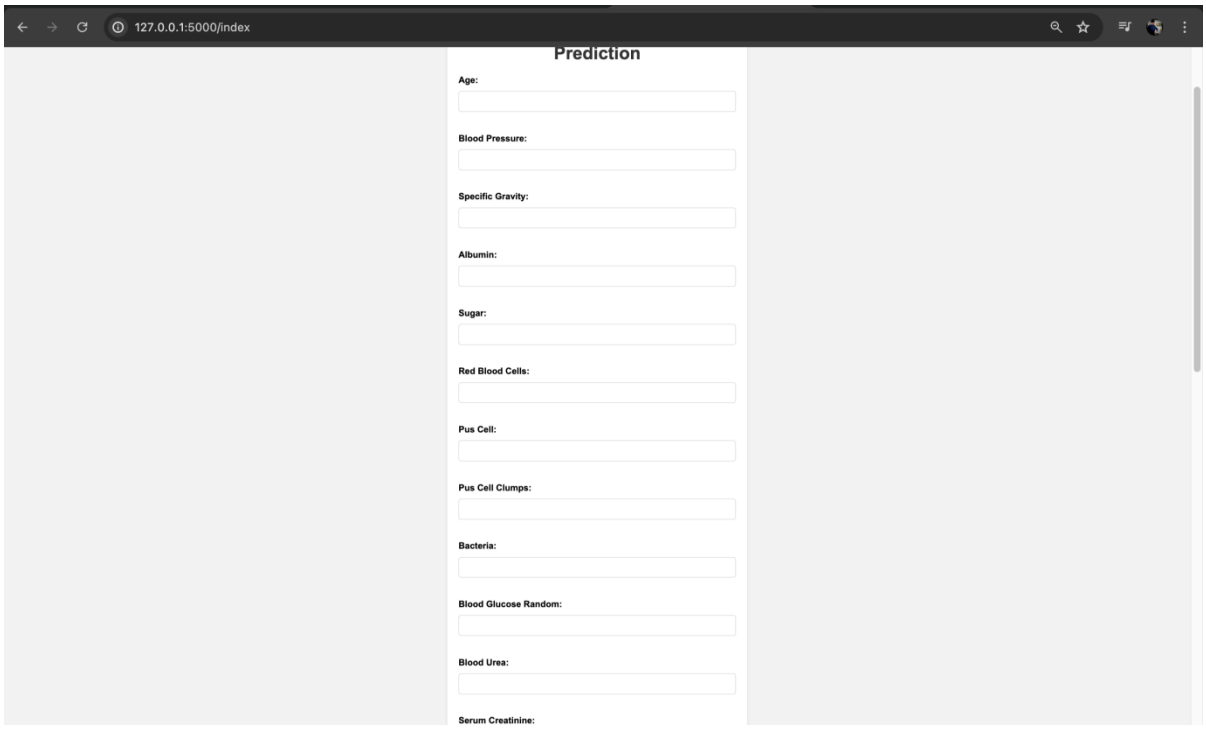
Nevertheless, Gradient Boosting models can be intricate, and tools such as, feature importance can be applied to determine which factors have the greatest influence on the models' decision-making. Such interpretability can be helpful for understanding possible tendencies in the on-time deliveries' determination.

6.Result:

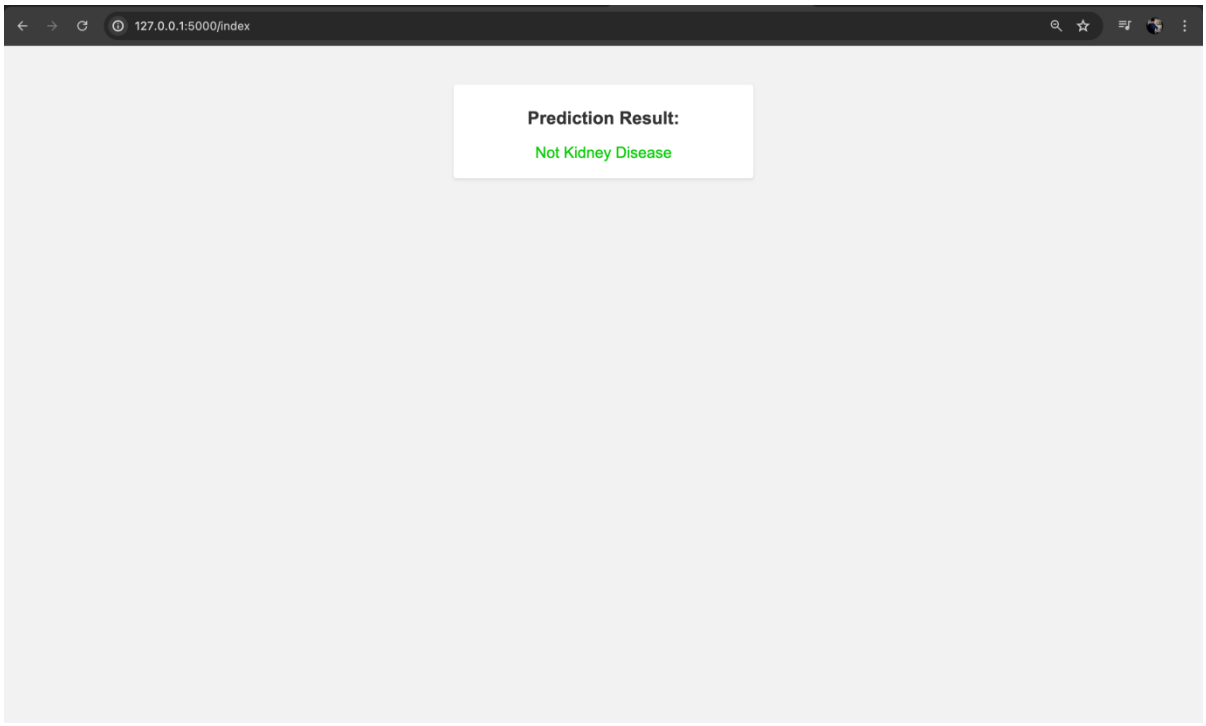
HomePage:



Index Page:



Result Page:



7 ADVANTAGES AND DISADVANTAGES

ADVANTAGES

● Integration with Flask:

Using Flask for integration is ideal due to its lightweight nature in Python web frameworks. It's versatile, simplifying the creation of user-friendly interfaces for CKD prediction models. Flask's straightforwardness accelerates model development and deployment significantly.

●Scalability:

Scalability is a strong suit of Flask-based models, designed to manage numerous concurrent user queries for real-time predictions effectively. This makes the model highly applicable in clinical settings with extensive patient populations, ensuring robust performance and scalability.

●Flexibility in Model Construction:

Combining a Voting Classifier with SVM, Decision Tree, KNN, and Naive Bayes models enables the exploitation of various algorithms' unique advantages while balancing their flaws. This ensemble method can raise the prediction model's overall robustness and accuracy.

●Interpretability:

Decision Tree and Naive Bayes models provide interpretable results, enabling medical practitioners to understand the rationale behind the models' predictions. This transparency boosts confidence and facilitates informed decision-making in clinical practice.

DISADVANTAGES

●Complexity of Model Building:

Selecting, training, and fine-tuning additional models is essential when incorporating them into a voting classifier. Achieving optimal performance requires careful optimization of each individual model and finding the right balance between them can be challenging.

●Model Overfitting:

Combining multiple models through voting increases the risk of overfitting, where a model may excel with training data but struggle to generalize to new data if not properly managed. To mitigate this risk, thorough cross-validation and regularization techniques are essential strategies.

●Computing Power:

Flask might require substantial computing resources to handle concurrent execution of multiple models, especially when dealing with complex training methods or extensive parameter spaces. Ensuring adequate hardware and infrastructure is essential to ensure smooth and efficient execution.

●Deployment and Upkeep:

Managing a Flask-based application entails ongoing monitoring, maintenance, and updates on the server. This can introduce additional complexity and potentially higher maintenance costs compared to deploying a standalone model.

8 APPLICATIONS :

Web-based CKD Risk Assessment Tool:

Users can enter their demographic and medical information via the intuitive web interface of the Flask application. The voting classifier model integrates predictions from SVM, Decision Tree, KNN, and Naive Bayes to estimate the probability of developing CKD. This functionality allows users to easily evaluate their CKD risk and make informed decisions to safeguard their health.

Clinical Decision Support System:

Integrating the Flask application into clinical settings can support healthcare providers in making informed decisions regarding the diagnosis and treatment of CKD. Utilizing a voting classifier model that amalgamates insights from diverse algorithms and patient data enables clinicians to enhance risk assessment, tailor therapy strategies, and facilitate early detection.

Programmes for Public Health Screening:

The Flask application is suitable for broader public health campaigns aimed at screening individuals for CKD risk. Community members can participate by entering their details into the system and promptly receive their CKD risk assessment. This approach facilitates more targeted preventive measures and educational initiatives tailored to high-risk groups.

Research and Data Analysis:

Flask can serve as a valuable tool for researchers and data analysts investigating CKD. They can leverage the application to predict CKD probabilities across large datasets, identifying critical factors, evaluating method effectiveness, and gaining deeper insights into the relationships between risk variables and CKD progression. This enables comprehensive exploration and understanding of CKD dynamics through advanced analytical approaches.

9 CONCLUSION :

In summary, deploying the prediction model using Flask and a voting classifier comprising Support Vector Machines (SVM), Decision Tree, K-Nearest Neighbours (KNN), and Naive Bayes algorithms offers a valuable resource for early detection of CKD. Healthcare professionals can effortlessly input patient data and obtain the CKD risk estimate derived from the combined forecasts of individual classifiers through integration into an intuitive web interface.

Integrating multiple classifiers in a voting ensemble harnesses the strengths of each method, resulting in a more dependable and accurate prediction. SVM, Decision Tree, KNN, and Naive Bayes contribute distinct perspectives and varied decision-making strategies, enhancing the overall effectiveness of the model.

The web application simplifies engagement with the prediction model through Flask, presenting healthcare professionals with a clear and comprehensible representation of the predicted CKD risk. This empowers them to make informed decisions regarding patient care and intervention strategies effectively.

Flask, the voting classifier, and the integrated algorithms work together to improve the model's early detection capabilities, enable personalised treatment plans, and improve patient outcomes in the management of CKD. Healthcare practitioners can use it as a usefultool to make effective decisions and allocate resources while addressing the global problem of chronic kidney disease.

10 FUTURE SCOPE:

The promise of machine learning (ML) models in the realm of chronic kidney disease (CKD) opens avenues for innovative research and practical application. Looking ahead, several potential pathways include...

Enhancing the accuracy and reliability of CKD prediction models remains an ongoing objective. Researchers strive to elevate prediction efficacy through exploration of advanced ML algorithms, ensemble strategies, and deep learning methodologies. Furthermore, integrating expansive and varied datasets, including real-time physiological and genomic data, holds promise for achieving superior predictive capabilities.

ML models can be advanced to monitor the progression of CKD and assess treatment efficacy longitudinally. Analyzing longitudinal data provides valuable insights into the evolving nature of CKD, empowering clinicians to tailor treatment plans according to individual patient trajectories. This approach also facilitates the identification of trends indicating improvement or deterioration of the condition over time.

ML models have the potential to stratify CKD patients into different risk categories based on factors such as disease severity, progression, and comorbidities. This stratification can aid in clinical decision support by enabling healthcare providers to optimize resource allocation, personalize treatment strategies, an

10. Appendix

10.1. Source Code : CLICK HERE

10.2. GitHub & Project Demo Link

GitHub : **CLICK HERE**

Project Demo Link: