CHRONIC KIDNEY DISEASE PREDICTION

Team Details
1. Sairy Nithya(20eg105440)
2. Tulasi Bhavana(20eg105447)
3. Eslavath Latha(20eg105409)

Project Supervisor
Dr. B. V. V. SIVA PRASAD
Associate Professor in CSE

Introduction

Chronic Kidney Disease is one of the most critical illness nowadays and proper diagnosis is required as soon as possible. Chronic kidney disease (CKD) is defined by persistent urine abnormalities, structural abnormalities or impaired excretory renal function suggestive of a loss of functional nephrons. The majority of patients with CKD are at risk of accelerated cardiovascular disease and death. Machine learning technique has become reliable for medical treatment. The massive quantities of data are analyzed using machine learning. It delivers faster and more accurate results in order to identify the risks.With the help of a machine learning classifier algorithms using features like age,blood pressure,urine specific gravity,albumin&sugar range,blood urea&glucose,hemoglobin,diabetes mellitus..etc the doctor can detect the disease on time.

Problem Statement

Developing machine learning-based prediction models for early Chronic Kidney Disease (CKD) detection, using a naïve bayes classifiers and feature selection methods, to enable timely diagnosis and improve patient outcomes
The main goal is to help doctors find out about CKD sooner, so they can treat it early and make patients feel better. By doing this, we want to make sure patients have a better chance of getting well.

Proposed Method

1.Data Collection:
Gather a dataset of patient records that includes relevant features such as age, gender, blood pressure, serum creatinine levels, blood glucose levels, family history of CKD, and other relevant medical information.

2.Data Preprocessing:
Clean and preprocess the data by handling missing values, normalizing numerical features, and encoding categorical variables. This step ensures that the data is in a suitable format for training the machine learning model.

3.Feature Selection :
Select specific columns (features) for analysis and drop some columns ('ba', 'sod', 'pot', 'appet', 'id', 'ane') that you don't want to use in your model.
Fill missing values in numerical columns with the mean.

Proposed Method

4.Data Splitting and Model Training :
Split the dataset into training and testing sets.
Train a machine learning model using Navie Bayes algorithm using the training data

5.Model Evaluation:
Evaluate the performance of the Naive Bayes model using the testing dataset. calculate accuracy and generate a classification report which includes metrics such as precision, recall, and F1-score .

Experiment Environment

- Programming Language : Python
- Machine Learning Libraries : Scikit-Learn
- Data Manipulation and Visualization Tools : Pandas, Matplotlib, Seaborn
- Code Development Environment : Google Colab

Experiment Screen shorts

```
[25] import pandas as pd
     import numpy as np
     from sklearn.model_selection import train_test_split
     from sklearn.naive_bayes import GaussianNB
     from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, classification_report,confusion_matrix
     from sklearn.preprocessing import StandardScaler
     import matplotlib.pyplot as plt
     import seaborn as sns
```

```
[26]
     data = pd.read_csv('/content/kidney_disease (1).csv')
     data
```

	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

```
[27]
     data = data.replace('?', np.nan)
```

```
[28]
     categorical_cols = ['rbc', 'pc', 'pcc', 'ba', 'htn', 'dm', 'cad', 'appet', 'pe', 'ane']
     data[categorical_cols] = data[categorical_cols].apply(lambda x: x.astype('category').cat.codes)
     data
```

Experiment Screen shorts

```
[29] numerical_cols = ['age', 'bp', 'sg', 'al', 'su', 'bgr', 'bu', 'sc', 'sod', 'pot', 'hemo', 'pcv', 'wc', 'rc']
     data[numerical_cols] = data[numerical_cols].apply(pd.to_numeric, errors='coerce')
     data[numerical_cols] = data[numerical_cols].fillna(data[numerical_cols].mean())
     data
```

```
[30]
     scaler = StandardScaler()
     data[numerical_cols] = scaler.fit_transform(data[numerical_cols])
```

```
[31] d=data.drop(['ba','sod','pot','appet','id','ane'],axis=1)
     d
```

```
[33] data.classification = data.classification.replace(to_replace='ckd\t',value='ckd')
```

```
[34] X=data.iloc[:,:-1]
     y=data.iloc[:,-1]
     X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
[35] nb_classifier = GaussianNB()
     nb_classifier.fit(X_train, y_train)
```

Experiment Screen shorts

```
[36]
y_pred = nb_classifier.predict(X_test)
```

```
[37]
accuracy = accuracy_score(y_test, y_pred)
```

```
report = classification_report(y_test, y_pred)
```

```
[41]
confusion_mat = confusion_matrix(y_test, y_pred)
```

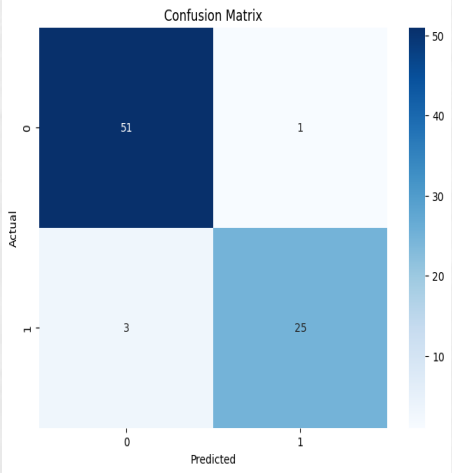
Experiment Results

```
[35]
GaussianNB
GaussianNB()
```

```
Classification Report:
              precision    recall  f1-score   support

        ckd       0.94      0.98      0.96        52
     notckd       0.96      0.89      0.93        28

    accuracy                           0.95        80
   macro avg       0.95      0.94      0.94        80
weighted avg       0.95      0.95      0.95        80
```




```
print(f"Accuracy: {accuracy}")
```

```
Accuracy: 0.95
```

Experiment Results

```python
import numpy as np
age = float(input("Enter age: "))
blood_pressure = float(input("Enter blood pressure: "))
specific_gravity = float(input("Enter specific gravity: "))
albumin = float(input("Enter albumin: "))
sugar = float(input("Enter sugar: "))
red_blood_cells = float(input("Enter red blood cells count: "))
pus_cells = float(input("Enter pus cells count: "))
pus_cell_clumps = input("Pus Cell Clumps (present/absent): ").lower()
bacteria = input("Bacteria (present/absent): ").lower()
hypertension = input("Hypertension (yes/no): ").lower()
diabetes = input("Diabetes (yes/no): ").lower()
appetite = input("Appetite (good/poor): ").lower()
anemia = input("Anemia (yes/no): ").lower()

user_data = np.array([[age, blood_pressure, specific_gravity, albumin, sugar, red_blood_cells, pus_cells,
                       pus_cell_clumps == "present", bacteria == "present",
                       hypertension == "yes", diabetes == "yes", appetite == "good", anemia == "yes"]])

prediction = clf.predict(user_data)

if prediction[0] == 1:
    print("You may be suffering from Chronic Kidney Disease (CKD). Please consult a healthcare professional.")
else:
    print("You are not predicted to have Chronic Kidney Disease (CKD).")
```

```
Enter age: 50
Enter blood pressure: 200
Enter specific gravity: 200
Enter albumin: 200
Enter sugar: 400
Enter red blood cells count: 300
Enter pus cells count: 300
Pus Cell Clumps (present/absent): 1
Bacteria (present/absent): 1
Hypertension (yes/no): 1
Diabetes (yes/no): 1
Appetite (good/poor): 0
Anemia (yes/no): 1
You are not predicted to have Chronic Kidney Disease (CKD).
```

Finding

We have discovered that Naive Bayes algorithm showed promising results in predicting chronic kidney disease, achieving a good level of accuracy in identifying patients with the condition.Handling and preprocessing of the data plays a crucial role in prediction.

Justification

1.What are parameters improved by our method

a) Prediction Accuracy

b) Performance

2.Mathametic formulas for calculating parameter values

Precision = TP / (TP + FP)

Recall = TP / (TP + FN)

F1-Score = 2 * (Precision * Recall) / (Precision + Recall)

3.why your parameter values improved?

Our parameters are improved using the preprocessing techniques such handling missing ,numerical and categorical data in the dataset.