Hope Artificial Intelligence

Classification Assignment

Problem Statement or Requirement:

A requirement from the Hospital, Management asked us to create a predictive model which will predict the Chronic Kidney Disease (CKD) based on the several parameters. The Client has provided the dataset of the same.

1.) Identify your problem statement

Problem Statement: Develop a predictive model to classify patients as having Chronic Kidney Disease (CKD) or not, based on provided medical parameters.

2.) Tell basic info about the dataset (Total number of rows, columns)

CKD Dataset Information:

- **Rows:** Usually around **400** records (varies by dataset).
- **Columns:** Around **25–30** medical parameters (e.g., age, blood pressure, sugar, albumin, hemoglobin).
- **Target Variable: CKD (Yes/No or 1/0)** indicating disease presence.
- **Data Types:** Mixture of **numerical** (age, blood pressure) and **categorical** (albumin, sugar levels).
- **Missing Values:** Common in medical datasets, requiring imputation.

3.) Mention the pre-processing method if you're doing any (like converting string to number – nominal data)

Pre-processing Steps:

- **Handle Missing Values:** Fill missing numerical values with mean/median and categorical values with mode.
- **Convert Categorical to Numerical:** Encode categorical features (e.g., 'normal' → 0, 'abnormal' → 1) using **Label Encoding or One-Hot Encoding**.
- **Feature Scaling:** Normalize continuous variables like age, blood pressure using **Min-Max Scaling or Standardization**.

5.) All the research values of each algorithm should be documented. (You can tabulate or screenshot of the results.)

```
print("The f1_macro value for best parameter{}:".format(grid.best_params_),f1_macro)
```

```
The f1_macro value for best parameter{'penalty': 'l2', 'solver': 'lbfgs'}: 0.9916844900066377
```

```
print("The confusion matrix:\n",cm)
```

```
The confusion matrix:
 [[45  0]
 [ 1 74]]
```

```
print("The classification report:\n",clf_report)
```

```
The classification report:
              precision    recall  f1-score   support

           0       0.98      1.00      0.99        45
           1       1.00      0.99      0.99        75

    accuracy                           0.99       120
   macro avg       0.99      0.99      0.99       120
weighted avg       0.99      0.99      0.99       120
```

```
from sklearn.metrics import roc_auc_score
roc_auc_score(y_test,grid.predict_proba(x_test)[:,1])
```

```
1.0
```

```
from sklearn.metrics import f1_score
f1_macro=f1_score(y_test,grid_prediction,average='weighted')
print("The f1_macro value for best parameter:".format(grid.best_params_),f1_macro)
```

```
The f1_macro value for best parameter: 0.9751481237656352
```

```
print("The confusion_matrix:\n",cm)
```

```
The confusion_matrix:
 [[45  0]
 [ 3 72]]
```

```
print("The classification_report:\n",clf_report)
```

```
The classification_report:
              precision    recall  f1-score   support

           0       0.94      1.00      0.97        45
           1       1.00      0.96      0.98        75

    accuracy                           0.97       120
   macro avg       0.97      0.98      0.97       120
weighted avg       0.98      0.97      0.98       120
```

```
from sklearn.metrics import roc_auc_score
roc_auc_score(y_test,grid.predict_proba(x_test)[:,1])
```

```
1.0
```

6.) Mention your final model, justify why you have chosen the same.

Final Model: Logistic Regression

Justification:

- **Simple & Easy to Interpret** for medical diagnosis.
- **Works Well for Binary Classification** like CKD (Yes/No).
- **Fast & Efficient** for smaller datasets.
- **Less Overfitting** compared to complex models.