Mathematical Calculations for IBD Model (GPR Model)

1. Xscaled=(X−μ)/σ

Where:

- X is the original value of the feature.
- μ is the mean of the feature.
- σ is the standard deviation of the feature.

2. $$k(x, x') = \exp\left(-\frac{\|x - x'\|^2}{2\ell^2}\right)$$

- `kernel = RBF(1.0, (1e-4, 1e3))`
- ℓ is **1.0** initially.
- k(x,x′) is the similarity (or 6kernel) between two input points x and x′.
- ‖x−x′‖^2 is the squared Euclidean distance between the two points x and x′: ‖x−x′‖^2=i=1∑n(xi−xi′)2 where xi and xi′ are the individual components of the vectors x and x′.
- $$\|x - x'\|^2 = \sum_{i=1}^{n} (x_i - x'_i)^2$$
- ℓ is the **length scale** parameter, which controls the width of the kernel. This is determined during the model training process, with bounds defined by (1e−4,1e3).

3. **Train-Test Split** (`train_test_split`)
The `train_test_split` function splits your dataset into training and testing datasets.

The formula for splitting the data into training and testing sets:

Xtrain,Xtest=train_test_split(Xencoded,yencoded,test_size=0.5,random_state=100)

Where:

- Xencoded is the feature matrix.
- yencoded is the target variable.
- The **test size** is 50% of the data (`test_size=0.5`), meaning you use 50% for training and 50% for testing.
- **`random_state=100`** is used to control the randomness of the data splitting.

4. Prediction and Thresolding

ypred,σpred=GPR.predict(Xtest,return_std=True)

Where:

- ypred are the predicted continuous values.

- σpred is the uncertainty (standard deviation) of the predictions.

Since we're performing **classification** (CKD vs. non-CKD), we apply a threshold of **0.7** to convert the continuous predictions into binary values (0 or 1):

$$y_{\text{pred\_class}} = \begin{cases} 1 & \text{if } y_{\text{pred}} > 0.7 \\ 0 & \text{if } y_{\text{pred}} \leq 0.7 \end{cases}$$

5. **Confusion Matrix**

The **confusion matrix** is computed using:

MatrixConfusion Matrix=confusion_matrix(ytest,ypred_class)

This matrix gives:

- True Positives (TP): Predicted 1, Actual 1
- True Negatives (TN): Predicted 0, Actual 0
- False Positives (FP): Predicted 1, Actual 0
- False Negatives (FN): Predicted 0, Actual 1

6. **Accuracy Calculation**

The **accuracy** is computed using:

Accuracy=(TP+TN)/(FP+FN+TP+TN)

Where:

- TP = True Positives
- TN = True Negatives
- FP = False Positives
- FN = False Negatives

7. **Accuracy per Point**

The **accuracy per point** is calculated by measuring how close each predicted value is to the actual value:

Accuracy per point=1− | ytest−ypred |