

## Project Development Phase Model Performance Test

Date	10 November 2022
Team ID	PNT2022TMID37589
Project Name	Project – Early Detection of Chronic Kidney Disease using Machine Learning
Maximum Marks	10 Marks

### Model Performance Testing:

Project team shall fill the following information in model performance testing template.

S.No.	Parameter	Values	Screenshot
1.	Metrics	<b>Regression Model:</b> MAE - , MSE - , RMSE - , R2 score -  <b>Classification Model:</b> Confusion Matrix - , Accuracy Score- & Classification Report -	See Below
2.	Tune the Model	Hyperparameter Tuning - Validation Method -	See Below

### 1. Metrics

#### Model: Random Forest Classification

**check model performance Random forest gives accurate predictions than logistic regression**

```
In [51]: accuracy_score(y_test,y_pred)
```

```
Out[51]: 0.95
```

```
In [52]: conf_mat=confusion_matrix(y_test,y_pred)
conf_mat
```

```
Out[52]: array([[52,  2],
               [ 2, 24]], dtype=int64)
```

```
In [53]: print(classification_report(y_test,y_pred))
```

	precision	recall	f1-score	support
0	0.96	0.96	0.96	54
1	0.92	0.92	0.92	26
accuracy			0.95	80
macro avg	0.94	0.94	0.94	80
weighted avg	0.95	0.95	0.95	80

```
In [54]: pickle.dump(lgr,open('CKD.pkl','wb'))
```

## 2. Tune the Model

### Hyperparameter Tuning:

- The number of features is important and should be tuned in random forest classification.
- Initially all parameters in the dataset are taken as independent values to arrive at the dependent decision of Chronic Kidney Disease or No Chronic Kidney Disease.
- But the result was not accurate so used only 8 more correlated values as independent values to arrive at the dependent decision of Chronic Kidney Disease or not.

### Validation Method:

It involves **partitioning the training data set into subsets, where one subset is held out to test the performance of the model**. This data set is called the validation data set.

Cross validation is to use different models and identify the best:

### Logistic Regression Model performance values:

#### check model performance Random forest gives accurate predictions than logistic regression

```
In [59]: accuracy_score(y_test,y_pred)
```

```
Out[59]: 0.925
```

```
In [60]: conf_mat=confusion_matrix(y_test,y_pred)
conf_mat
```

```
Out[60]: array([[48,  6],
               [ 0, 26]], dtype=int64)
```

```
In [61]: print(classification_report(y_test,y_pred))
```

	precision	recall	f1-score	support
0	1.00	0.89	0.94	54
1	0.81	1.00	0.90	26
accuracy			0.93	80
macro avg	0.91	0.94	0.92	80
weighted avg	0.94	0.93	0.93	80

```
In [54]: pickle.dump(lgr,open('CKD.pkl','wb'))
```

Hence we tested with Logistic regression and Random Forest Classification wherein the accuracy of Random Forest classification is 95% compared with Logistic Regression.

Metric	Logistic Regression	Random Forest Classification																																																												
Accuracy	0.925	0.95																																																												
Other metrics	<pre>accuracy_score(y_test,y_pred)</pre> <p>0.925</p> <pre>conf_mat=confusion_matrix(y_test,y_pred)</pre> <pre>conf_mat</pre> <pre>array([[48, 6],</pre> <pre>       [ 0, 26]], dtype=int64)</pre> <pre>print(classification_report(y_test,y_pred))</pre> <table><thead><tr><th></th><th>precision</th><th>recall</th><th>f1-score</th><th>support</th></tr></thead><tbody><tr><td>0</td><td>1.00</td><td>0.89</td><td>0.94</td><td>54</td></tr><tr><td>1</td><td>0.81</td><td>1.00</td><td>0.90</td><td>26</td></tr><tr><td>accuracy</td><td></td><td></td><td>0.93</td><td>80</td></tr><tr><td>macro avg</td><td>0.91</td><td>0.94</td><td>0.92</td><td>80</td></tr><tr><td>weighted avg</td><td>0.94</td><td>0.93</td><td>0.93</td><td>80</td></tr></tbody></table>		precision	recall	f1-score	support	0	1.00	0.89	0.94	54	1	0.81	1.00	0.90	26	accuracy			0.93	80	macro avg	0.91	0.94	0.92	80	weighted avg	0.94	0.93	0.93	80	<pre>: accuracy_score(y_test,y_pred)</pre> <p>: 0.95</p> <pre>: conf_mat=confusion_matrix(y_test,y_pred)</pre> <pre>conf_mat</pre> <pre>: array([[52, 2],</pre> <pre>        [ 2, 24]], dtype=int64)</pre> <pre>: print(classification_report(y_test,y_pred))</pre> <table><thead><tr><th></th><th>precision</th><th>recall</th><th>f1-score</th><th>support</th></tr></thead><tbody><tr><td>0</td><td>0.96</td><td>0.96</td><td>0.96</td><td>54</td></tr><tr><td>1</td><td>0.92</td><td>0.92</td><td>0.92</td><td>26</td></tr><tr><td>accuracy</td><td></td><td></td><td>0.95</td><td>80</td></tr><tr><td>macro avg</td><td>0.94</td><td>0.94</td><td>0.94</td><td>80</td></tr><tr><td>weighted avg</td><td>0.95</td><td>0.95</td><td>0.95</td><td>80</td></tr></tbody></table>		precision	recall	f1-score	support	0	0.96	0.96	0.96	54	1	0.92	0.92	0.92	26	accuracy			0.95	80	macro avg	0.94	0.94	0.94	80	weighted avg	0.95	0.95	0.95	80
	precision	recall	f1-score	support																																																										
0	1.00	0.89	0.94	54																																																										
1	0.81	1.00	0.90	26																																																										
accuracy			0.93	80																																																										
macro avg	0.91	0.94	0.92	80																																																										
weighted avg	0.94	0.93	0.93	80																																																										
	precision	recall	f1-score	support																																																										
0	0.96	0.96	0.96	54																																																										
1	0.92	0.92	0.92	26																																																										
accuracy			0.95	80																																																										
macro avg	0.94	0.94	0.94	80																																																										
weighted avg	0.95	0.95	0.95	80																																																										

The above table shows that Random Forest Classification gives better results over Logistic Regression.