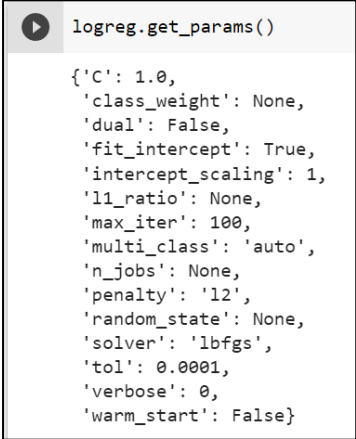
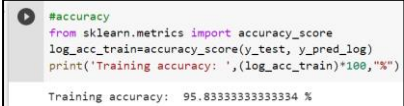
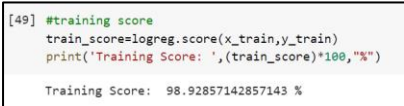
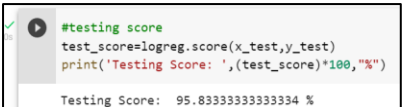


## Project Development Phase Model Performance Test

Date	10 November 2022
Team ID	PNT2022TMID36002
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	Screenshots
1.	Model Summary	{'C': 1.0, 'class_weight': None, 'dual': False, 'fit_intercept': True, 'intercept_scaling': 1, 'l1_ratio': None, 'max_iter': 100, 'multi_class': 'auto', 'n_jobs': None, 'penalty': 'l2', 'random_state': None, 'solver': 'lbfgs', 'tol': 0.0001, 'verbose': 0, 'warm_start': False}	 <pre>logreg.get_params()  {'C': 1.0,  'class_weight': None,  'dual': False,  'fit_intercept': True,  'intercept_scaling': 1,  'l1_ratio': None,  'max_iter': 100,  'multi_class': 'auto',  'n_jobs': None,  'penalty': 'l2',  'random_state': None,  'solver': 'lbfgs',  'tol': 0.0001,  'verbose': 0,  'warm_start': False}</pre>
2.	Accuracy and Scores	<p><b>Training Accuracy:</b> 95.83333333333334 %</p> <p><b>Training Score:</b> 98.92857142857143 %</p> <p><b>Testing Score:</b> 95.83333333333334 %</p>	<p><b>Training Accuracy:</b></p>  <pre>#accuracy from sklearn.metrics import accuracy_score log_acc_train=accuracy_score(y_test, y_pred_log) print('Training accuracy: ',(log_acc_train)*100,"%")  Training accuracy: 95.83333333333334 %</pre> <p><b>Training Score:</b></p>  <pre>[49] #training score train_score=logreg.score(x_train,y_train) print('Training Score: ',(train_score)*100,"%")  Training Score: 98.92857142857143 %</pre> <p><b>Testing Score:</b></p>  <pre>#testing score test_score=logreg.score(x_test,y_test) print('Testing Score: ',(test_score)*100,"%")  Testing Score: 95.83333333333334 %</pre>

3.	Metrics	<p><b>Classification Report:</b></p> <p>It contains the precision, recall, F1-score, and support of the logistic regression model</p> <p><b>Regression Metrics:</b></p> <p>-&gt; <b>MAE:</b> 0.041666666666666664</p> <p>-&gt; <b>MSE:</b> 0.2041241452319315</p> <p>-&gt; <b>RMSE:</b> 0.45180100180492244</p> <p><b>Confusion Matrix</b></p> <p><b>ROC Curve:</b></p> <p>A graph to show the performance of a classification model at all classification thresholds.</p>	<p><b>Classification Report:</b></p> <pre>print("Classification Report:\n", clsrep_log)</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>0.99</td><td>0.95</td><td>0.97</td><td>78</td></tr><tr><td>1</td><td>0.91</td><td>0.98</td><td>0.94</td><td>42</td></tr><tr><td>accuracy</td><td></td><td></td><td>0.96</td><td>120</td></tr><tr><td>macro avg</td><td>0.95</td><td>0.96</td><td>0.95</td><td>120</td></tr><tr><td>weighted avg</td><td>0.96</td><td>0.96</td><td>0.96</td><td>120</td></tr></tbody></table> <p><b>Regression Metrics:</b></p> <pre>[55] print("Regression Metrics: ") print("MAE : ",mae_log) print("MSE : ",mse_log) print("RMSE : ",rmse_log)</pre> <p>Regression Metrics:</p> <table><tbody><tr><td>MAE</td><td>: 0.041666666666666664</td></tr><tr><td>MSE</td><td>: 0.2041241452319315</td></tr><tr><td>RMSE</td><td>: 0.45180100180492244</td></tr></tbody></table> <p><b>Confusion Matrix:</b></p> <p><b>ROC Curve:</b></p>		precision	recall	f1-score	support	0	0.99	0.95	0.97	78	1	0.91	0.98	0.94	42	accuracy			0.96	120	macro avg	0.95	0.96	0.95	120	weighted avg	0.96	0.96	0.96	120	MAE	: 0.041666666666666664	MSE	: 0.2041241452319315	RMSE	: 0.45180100180492244
	precision	recall	f1-score	support																																			
0	0.99	0.95	0.97	78																																			
1	0.91	0.98	0.94	42																																			
accuracy			0.96	120																																			
macro avg	0.95	0.96	0.95	120																																			
weighted avg	0.96	0.96	0.96	120																																			
MAE	: 0.041666666666666664																																						
MSE	: 0.2041241452319315																																						
RMSE	: 0.45180100180492244																																						
4.	Tune the Model	<p><b>Hyperparameter Tuning:</b></p> <p>Tuning the hyperparameters of logistic regression using GridSearchCV.</p>	<pre>[67] from sklearn.model_selection import GridSearchCV import warnings warnings.filterwarnings('ignore') # parameter grid parameters = {     'penalty': ['l1','l2'],     'C': np.logspace(-3,3,7),     'solver': ['newton-cg', 'lbfgs', 'liblinear'], }  [68] logreg1 = LogisticRegression() clf = GridSearchCV(logreg1, # model                    param_grid = parameters, # hyperparameters                    scoring='accuracy', # metric for scoring                    cv=10) # number of folds  [69] clf.fit(x_train,y_train)  print("Tuned Hyperparameters :", clf.best_params_) print("Accuracy :",clf.best_score_) Tuned Hyperparameters : {'C': 1.0, 'penalty': 'l2', 'solver': 'lbfgs'} Accuracy : 0.975  [66] logreg1 = LogisticRegression(C = 1.0,                                 penalty = 'l2',                                 solver = 'liblinear') logreg1.fit(x_train,y_train) y_pred = logreg1.predict(x_test) print("Accuracy:",logreg1.score(x_test, y_test))  Accuracy: 0.9583333333333334</pre>																																				

