CLASSIFICATION ASSIGNMENT

1)PROBLEM STATEMENT :

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.

Machine Learning -Supervised Learning -Classification

Name of the project :Chronic kidney disease prediction

2)DATASET

Total number of rows :25
Total number of columns :400

3) Nominal Data

converting String to number

4)

1)Decision Tree Classification

param_max_features	param_splitter	params	split0_test_score	split1_test_score	split2_test_score	mean_test_score	std_test_score	rank_test_score
auto	best	{'criterion': 'gini', 'max_features': 'auto', ...	0.966392	0.922498	0.953877	0.947565	0.018494	11
auto	random	{'criterion': 'gini', 'max_features': 'auto', ...	0.977528	0.955309	0.954241	0.962390	0.010743	6
sqrt	best	{'criterion': 'gini', 'max_features': 'sqrt', ...	0.966182	0.922498	0.954241	0.947615	0.018465	10
sqrt	random	{'criterion': 'gini', 'max_features': 'sqrt', ...	0.988797	0.977528	0.988669	0.984984	0.005287	1
log2	best	{'criterion': 'gini', 'max_features': 'log2', ...	0.966392	0.933485	0.965550	0.955103	0.015333	7
		{'criterion': 'gini'						

2.LOGISTIC ALGORITHM

td_score_time	param_penalty	param_solver	params	split0_test_score	split1_test_score	split2_test_score	mean_test_score	std_test_score	rank_test_score
0.004956	l2	newton-cg	{'penalty': 'l2', 'solver': 'newton-cg'}	0.977654	0.988797	1.000000	0.988775	0.009114	1
0.002779	l2	lbfgs	{'penalty': 'l2', 'solver': 'lbfgs'}	0.977654	0.988797	1.000000	0.988775	0.009114	1
0.001854	l2	liblinear	{'penalty': 'l2', 'solver': 'liblinear'}	0.966561	0.966561	0.977397	0.970146	0.005098	4
0.000902	l2	saga	{'penalty': 'l2', 'solver': 'saga'}	0.977654	0.966561	0.988669	0.977587	0.009017	3

3.RANDOM FOREST CLASSIFICATION

ime	param_criterion	param_max_features	params	split0_test_score	split1_test_score	split2_test_score	mean_test_score	std_test_score	rank_test_score
383	gini	auto	{'criterion': 'gini', 'max_features': 'auto'}	0.966392	0.933485	0.977121	0.958931	0.018567	6
372	gini	sqrt	{'criterion': 'gini', 'max_features': 'sqrt'}	0.966392	0.944486	0.977121	0.962612	0.013578	3
462	gini	log2	{'criterion': 'gini', 'max_features': 'log2'}	0.977528	0.944486	0.965550	0.962510	0.013683	4
068	entropy	auto	{'criterion': 'entropy', 'max_features': 'auto'}	1.000000	0.944486	0.988600	0.977654	0.023975	1
947	entropy	sqrt	{'criterion': 'entropy', 'max_features': 'sqrt'}	0.988727	0.955508	0.977121	0.973773	0.013790	2
038	entropy	log2	{'criterion': 'entropy', 'max_features': 'log2'}	0.977528	0.922498	0.977121	0.958981	0.025871	5

SUPPORT VECTOR MACHINE

n_gamma	param_kernel	params	split0_test_score	split1_test_score	split2_test_score	split3_test_score	split4_test_score	mean_test_score	std_test_score	rank_test_score
auto	linear	{'C': 10, 'gamma': 'auto', 'kernel': 'linear'}	0.981569	0.963284	0.962573	0.981031	0.980868	0.973828	0.008944	23
auto	rbf	{'C': 10, 'gamma': 'auto', 'kernel': 'rbf'}	0.963284	0.981378	1.000000	1.000000	1.000000	0.988766	0.014760	5
auto	poly	{'C': 10, 'gamma': 'auto', 'kernel': 'poly'}	1.000000	0.981378	1.000000	1.000000	0.980868	0.992480	0.009241	1
		{'C': 10, 'gamma':								

Support Vector Machine gives best Accuracy :0.99