_	fit_transform(Te_Label)
<pre>def IELm_objective(dimens) a = dimensions[0:8].r b=dimensions[-1].resh predicted_accuracy=0 beta=0 mul=np.dot(Train_feat) sum=mul+b activation=relu(sum) beta_overall=np.dot(r keeps on changing with r y=np.dot(activation,k E=E-y # Updated Train predicted_rmse=RMSE(y)</pre>	<pre>lared_error(y_actual, y_predicted)) sions,E): reshape((len(Train_features[0]),1)) chape(1,1) #Initial value of accuracy stures,a) # activation function sigmoid to calculate H np.linalg.pinv(activation),E) # beta= inverse(H) * Target Label. As our new neuron</pre>
# Main Algorithm that indive the best possible # perfromance with the add L=0 # Number of neurons of max_neuron=50 # Maximum of max_neuron=50 # Maximum of max_neurons RMSE_expected=0.05 #Expended the neurons predicted_rmse=100 #Indicated the target and the predicted new_a=np.random.uniform(1) ts for the first hidden of new_b=np.random.uniform(1) print(new_a) # ***** TRAININIG STARTS #Loop until the hidden lates surpass the expected accession to the start_time=time.time() options = {'c1': 0.5, 'c2'}	<pre>put,**kwargs): dape[0] cles): dctive(input[i],**kwargs) dcreases the value of neurons in the hidden layer one by one and trying dded neurons in hidden layer inititally number of Neurons bected rmse or a threshold value that the network should have to break so initial value of rmse value of error as training set label, to keep a check on the differnce dicted low=-1,high=1,size=[len(Train_features[0]),1]) #Generating numpy array neuron low=-1,high=1,size=[1]) TS HERE ********* ayer reaches maximum number of neurons and until the predicted accuracy ccuracy 22': 0.3, 'w':0.9} #Setting the hyper parameter and passing to the optin -1,-1,-1] #setting the upper and lower limit bounds for the weights and</pre>
<pre>if L==1: mul=np.dot(Train_sum=mul+new_b activation=relu(sbeta_overall=np.cy=np.dot(activative=E-ypredicted_rmse=RNprint("</pre>	
print(pos) a=pos[:8].reshape # a=np.random.ung ghts for the second neuro # b=np.random.ung and so on b=pos[-1].reshape new_a=np.append(r mul=np.dot(Train_ new_b=np.append(r sum=mul+b activation=relu(s) beta=np.dot(pinv2 y=np.dot(activation E=E-y beta_overall=np.append(r print(predicted_rend_time=round(time.time) print("TRAINING TIME",end	<pre>diform(low=-1,high=1,size=[1]) #creating a new random bias for the second e(1,) new_a,a,axis=1) a_features,a) new_b,b,axis=0) sum) 22(activation), E) dion,beta) append(beta_overall,beta,axis=0) append(by,Train_Label) rmse,"with", L," Neuron ") e()-start_time,3)</pre>
'c2': 0.3, 'w': 0.9} pyswarms.single.global_be 6/50, best_cost=0.18 [[-0.66107378] [-0.32275013] [-0.01174512] [-0.64901455] [-0.68621937] [0.01196192] [-0.0977581] [0.56662726]] TRAINING RMSE 0.45298840411497093 With pyswarms.single.global_be 0, best_cost=0.166	. 1 Neuron
-0.26347853 0.57193597 2020-10-30 22:02:40,439 - 'c2': 0.3, 'w': 0.9} pyswarms.single.global_be 0, best_cost=0.138 [0.32740918 0.35734395 -0.26347853 0.57193597 0.43067502740785935 with pyswarms.single.global_be 0, best_cost=0.138 2020-10-30 22:02:40,765 - 375174565345297, best pos 0.23366358 0.51934679 2020-10-30 22:02:40,778 - 'c2': 0.3, 'w': 0.9} pyswarms.single.global_be 0, best_cost=0.137	- pyswarms.single.global_best - INFO - Optimize for 50 iters with {'c1'est: 34%
0.45298840411497093 with pyswarms.single.global_be 0, best_cost=0.137 2020-10-30 22:02:41,088 - 3713167498343778, best po -0.67473244 0.07933197 2020-10-30 22:02:41,098 - 'c2': 0.3, 'w': 0.9} pyswarms.single.global_be 0, best_cost=0.167 [-0.17879421 0.13136125 -0.67473244 0.07933197 0.45009710681289755 with pyswarms.single.global_be 0, best_cost=0.155 2020-10-30 22:02:41,404 -	est: 100%
-0.65750547 -0.68582234 2020-10-30 22:02:41,416 - 'c2': 0.3, 'w': 0.9} pyswarms.single.global_be 0, best_cost=0.254 [0.28596516 0.58846337 -0.65750547 -0.68582234 0.3878346363334901 with 5 pyswarms.single.global_be 0, best_cost=0.239 2020-10-30 22:02:41,722 - 3863191556026067, best po 0.20424368 -0.52439884 2020-10-30 22:02:41,732 - 'c2': 0.3, 'w': 0.9} pyswarms.single.global_be 50, best_cost=0.212	0.03045099] - pyswarms.single.global_best - INFO - Optimize for 50 iters with {'c1'est: 34%
1209841325265055, best pool 0.06801675 -0.65053353 2020-10-30 22:02:42,065 -0.62': 0.3, 'w': 0.9} pyswarms.single.global_best pool 0.06801675 -0.65053353 0.45298840411497093 with pyswarms.single.global_best pool 0.207 2020-10-30 22:02:42,362 -0.53625536 0.95875317 2020-10-30 22:02:42,374 -0.53625526 0.95875317 2020-10-30 22:02:42,374 -0.5362526 0.95875317 2020-10-30 22:02:42,374 -0.5362526 0.95875317 2020-10-30 22:02:42,374 -0.55626 0.958752 0.958752 0.958752 0.958752 0.958752 0.958752 0.958752 0.958752 0.958752 0.958752 0.958752 0.958752 0.958752 0.958752 0.958752 0.958752 0.958752	- pyswarms.single.global_best - INFO - Optimize for 50 iters with {'c1'est: 36%
-0.53625536 0.95875317 0.2657276901476472 with 8 pyswarms.single.global_be 0, best_cost=0.372 2020-10-30 22:02:42,676 -7239270845141437, best po -0.23196537 -0.7816018 2020-10-30 22:02:42,688 -1c2': 0.3, 'w': 0.9} pyswarms.single.global_be 0, best_cost=0.392	0.72111647 0.84548997 0.97089818 0.46756097 -0.55684406] 8 Neuron est: 100%
0, best_cost=0.375 2020-10-30 22:02:43,023 - 754519678188084, best pos 0.14467538 0.83696637 2020-10-30 22:02:43,040 - 'c2': 0.3, 'w': 0.9} pyswarms.single.global_bes 50, best_cost=0.395 [-0.55785016 0.56417514 0.14467538 0.83696637 0.505277420327198 with 10 pyswarms.single.global_bes 0, best_cost=0.382 2020-10-30 22:02:43,375 - 8224358315404905, best pos 0.66415534 0.77237805 2020-10-30 22:02:43,391 - 'c2': 0.3, 'w': 0.9}	- pyswarms.single.global_best - INFO - Optimization finished best coss: [-0.55785016 0.56417514 0.58957993 0.2688383 -0.73715879 0.1066 0.33362569] - pyswarms.single.global_best - INFO - Optimize for 50 iters with {'c1'est: 32%
'c2': 0.3, 'w': 0.9} pyswarms.single.global_be 50, best_cost=0.331 [-0.10373958 0.14040155 0.66415534 0.77237805 0.4281493310406408 with 1 pyswarms.single.global_be 0, best_cost=0.331 2020-10-30 22:02:43,773 - 313384468607611, best pos 0.8043446 0.07433328 0. 2020-10-30 22:02:43,786 - 'c2': 0.3, 'w': 0.9} pyswarms.single.global_be 0, best_cost=0.402 [0.00184292 0.91848331 0. 0.8043446 0.07433328 0.	est: 28%
0.42278782840457635 with pyswarms.single.global_be 0, best_cost=0.395 2020-10-30 22:02:44,175 - 9501946040405633, best po 0.5810754 -0.62154789 2020-10-30 22:02:44,188 - 'c2': 0.3, 'w': 0.9} pyswarms.single.global_be 0, best_cost=0.423 [0.92499365 0.00652414 0.5810754 -0.62154789 0.42078734109399124 with pyswarms.single.global_be 0, best_cost=0.423 2020-10-30 22:02:44,507 - 227646410178939, best pos -0.34596992 0.50826365	est: 100%
'c2': 0.3, 'w': 0.9} pyswarms.single.global_be 50, best_cost=0.354 [-0.88824103	est: 32% 0.75575011 0.15570277 0.10062004 0.94101642 0.0469956] 14 Neuron est: 100% 0.15280895 0.46466163 -0.94361899 0.51123665 -0.42902773 -0.59691279 -0.31739605] - pyswarms.single.global_best - INFO - Optimize for 50 iters with {'c1'} est: 32% 0.51123665 -0.42902773 -0.59691279 -0.31739605]
0.45427056358286394 with pyswarms.single.global_be 0, best_cost=0.372 2020-10-30 22:02:45,168 - 717733357271424, best pos -0.47079046 -0.20793457 2020-10-30 22:02:45,180 - 'c2': 0.3, 'w': 0.9} pyswarms.single.global_be 50, best_cost=0.394 [0.11587772	est: 100%
50, best_cost=0.437 [-0.5228362 -0.05046581 0.54765862 0.94494234 0.4217353157329687 with 1 pyswarms.single.global_be 0, best_cost=0.435 2020-10-30 22:02:45,808 -3496821153978504, best po -0.16268157 0.99096167 2020-10-30 22:02:45,818 -102': 0.3, 'w': 0.9} pyswarms.single.global_be 0, best_cost=0.401 [-0.05921354 -0.04638522 -0.16268157 0.99096167 0.45534775335473604 with	-0.61997774 -0.541295 -0.04010865 -0.09142829 0.15093844] 17 Neuron est: 100% - pyswarms.single.global_best - INFO - Optimization finished best cosos: [-0.05921354 -0.04638522 0.19105752 0.32381809 0.20085224 -0.752-0.23528601] - pyswarms.single.global_best - INFO - Optimize for 50 iters with {'c1'est: 34% 0.19105752 0.32381809 0.20085224 -0.75288083 -0.23528601] 18 Neuron
pyswarms.single.global_be 0, best_cost=0.401 2020-10-30 22:02:46,128 - 0081963867445713, best po -0.72431554 0.3886159 2020-10-30 22:02:46,139 - 'c2': 0.3, 'w': 0.9} pyswarms.single.global_be 0, best_cost=0.388 [0.81375338	- pyswarms.single.global_best - INFO - Optimization finished best costos: [0.81375338 0.38452979 -0.57740213 0.22647946 0.8874637 -0.2540.07621042] - pyswarms.single.global_best - INFO - Optimize for 50 iters with {'c1'est: 34%
[-0.11081391 0.11312379 0.64357328 -0.27595752 0.41019516865560857 with pyswarms.single.global_be 0, best_cost=0.427 2020-10-30 22:02:46,796 -2715023725786005, best po -0.30107604 -0.07052459 2020-10-30 22:02:46,806 -1c2': 0.3, 'w': 0.9} pyswarms.single.global_be 0, best_cost=0.376	-0.29382293] 20 Neuron est: 100%
0.5035061328682183 with 2 pyswarms.single.global_be 0, best_cost=0.376 2020-10-30 22:02:47,123 - 7610167751392376, best po 0.9159617 -0.93752817 2020-10-30 22:02:47,133 - 'c2': 0.3, 'w': 0.9} pyswarms.single.global_be 50, best_cost=0.367 [-0.29109469 -0.35104246 0.9159617 -0.93752817 0.45298840411497093 with pyswarms.single.global_be 0, best_cost=0.367 2020-10-30 22:02:47,468 -	21 Neuron est: 100%
0.00419631 -0.21306663 2020-10-30 22:02:47,477 - 'c2': 0.3, 'w': 0.9} pyswarms.single.global_be 6/50, best_cost=0.39 [0.01147836 0.02861879 0.00419631 -0.21306663 0.4331791767375511 with 2 pyswarms.single.global_be 0, best_cost=0.381 2020-10-30 22:02:47,818 - 8120346333995087, best po 0.69882426 0.45787489	- pyswarms.single.global_best - INFO - Optimize for 50 iters with {'c1'est: 32%
50, best_cost=0.428	24 Neuron est: 100%
pyswarms.single.global_be 50, best_cost=0.428 [0.92440131 0.82479814	25 Neuron est: 100% 1100 1100 1100 1100 110 110
pyswarms.single.global_best	0.71206606] 25 Neuron est: 100%
pyswarms.single.global_be 50, best_cost=0.428 [0.92440131 0.82479814	0.71206606) 25 Neuron est: 100%
pyswarms.single.global_be 50, best_cost=0.428 [0.92440131	0.7;206806 25 Neuron est: 1008
pyswarms.single.global_be 50, best_cost=0.428 [0.92440131	0.71206506] - pyswarms.single.global_bear - TNTO - Optimization indished bear cost: 1008 - pyswarms.single.global_bear - INYO - Optimization indished bear cost: 1002 - pyswarms.single.global_bear - INYO - Optimization indished bear cost: 30% - pyswarms.single.global_bear - INYO - Optimization indished bear cost: 100% - pyswarms.single.global_bear - INYO - Optimization indished bear cost: 100% - pyswarms.single.global_bear - INYO - Optimization indished bear cost: 100% - pyswarms.single.global_bear - INYO - Optimization indished bear cost: 100% - pyswarms.single.global_bear - INYO - Optimization indished bear cost: 100% - pyswarms.single.global_bear - INYO - Optimization indished bear cost: 100% - pyswarms.single.global_bear - INYO - Optimization Indished bear cost: 100% - pyswarms.single.global_bear - INYO - Optimization Indished bear cost: 100% - pyswarms.single.global_bear - INYO - Optimization Indished bear cost: 100% - pyswarms.single.global_bear - INYO - Optimization Indished bear cost: 120% - pyswarms.single.global_bear - INYO - Optimization Indished bear cost: 120% - pyswarms.single.global_bear - INYO - Optimization Indished bear cost: 100% - pyswarms.single.global_bear - INYO - Optimization Indished bear cost: 100% - pyswarms.single.global_bear - INYO - Optimization Indished bear cost: 100% - pyswarms.single.global_bear - INYO - Optimization Indished bear cost: 100% - pyswarms.single.global_bear - INYO - Optimization Indished bear cost: 100% - pyswarms.single.global_bear - INYO - Optimization Indished bear cost: 100% - pyswarms.single.global_bear - INYO - Optimization Indished bear cost: 100% - pyswarms.single.global_bear - INYO - Optimization Indished bear cost: 100% - pyswarms.single.global_bear - INYO - Optimization Indished bear cost: 100% - pyswarms.single.global_bear - INYO - Optimization Indished bear cost: 100% - pyswarms.single.global_bear - INYO - Optimization Indished bear cost: 100% - pyswarms.single.global_bear - INY
pyswarms.single.global_be 50, best_cost=0.428 [0.9244013	### 1995 Common ### 1995 Common ### 1995 Common ### 1995 Common
pyswarms.single.global_be 50, best_cost=0.428 [0.92440131	0.711000001 0.71100001 0.71100001 0.71100
yyswarms.single.global_be 50, best_cost=0.428 10.69882426 0.4578748 0.42878213855705 with 2 yyswarms.single.global_be 0.42878213855705 with 2 yyswarms.single.global_be 0.5782343 0.45878429 2020-10-30 22:02:48,160- 0.5782343 -0.58796329 2020-10-30 22:02:48,160- 0.5782343 -0.58796329 2020-10-30 22:02:48,160- 0.5782343 -0.58796329 2020-10-30 22:02:68,160- 0.5782343 -0.58796329 2020-10-30 22:02:68,160- 0.5782343 -0.58796329 0.45842620464604 0.041625 0.07789316 -0.06482388 -0.5782343 -0.58796329 0.45842620464636275 with yyswarms.single.global_be 0.best_cost=0.415 0.2020-10-30 22:02:48,482- 15198872272343, best po- 0.24204694 0.041625 0.4564632270989616 with 2 yyswarms.single.global_be 0.best_cost=0.418 0.2020-10-30 2:02:02:48,860- 80.203003195 0.7608066 0.4564632270989616 with 2 yyswarms.single.global_be 0.best_cost=0.42 0.2010-30 2:02:02:49,870- 0.2210-30 2:02:02:49,870- 0.2210-30 2:02:02:49,870- 0.2210-30 2:02:03,980- 0.25003695 0.7608066 0.44162787433992376 with yyswarms.single.global_be 0.best_cost=0.41 2020-10-30 2:02:49,201- 0.230030195 0.70800666 0.44162787433992376 with yyswarms.single.global_be 0.best_cost=0.41 2020-10-30 2:02:49,201- 0.4428719384022625 with yyswarms.single.global_be 0.best_cost=0.41 2020-10-30 2:02:49,542- 9943645786602, best pos: 5.501785726-01] 0.4428719384022625 with yyswarms.single.global_be 0.best_cost=0.42 2020-10-30 2:02:49,542- 9943645798602, best pos: 0.8607664 0.3860016 0.8607664 0.3860016 0.8607664 0.3860016 0.8607664 0.3860016 0.8607664 0.3860016 0.8607664 0.3860016 0.8607664 0.3860016 0.08076865059 0.99362495 0.2020-10-30 2:02:49,542- 9943645798602, best pos: 0.2020-10-30 2:02:49,542- 9943645798602, best pos: 0.80219457 -0.50847604 0.90210-30 2:02:50,684- 0.2090000000000000000000000000000000000	0.1013/00000 ***COLOR Comparison of Color
pyswarms.single.global per both best_cost=0.48247814	20.00000000000000000000000000000000000
yswarms.single.global be yest-cost-0.424.04.0. 10.9244031-0.824.9914 0.0.92912420-0.424.9914 0.0.92912420-0.424.9914 0.0.92912420-0.424.9919 0.0.921230-0.428.1000-1.000 0.0.921230-0.200.2000-1.000 0.0.921230-0.200.200.200.1000 0.0.920230-0.200.200.200.1000 10.007693130-0.06842388 -0.57023431-0.05842388 -0.5702348-0.05842388 -0.5702348-0.05842388 -0.5702348-0.05842388 -0.5702348-0.05842388 -0.5702348-0.05842388 -0.5702348-0.05842388 -0.5702348-0.05842388 -0.5702348-0.05842388 -0.5702348-0.05842388 -0.5702348-0.05842388 -0.5702348-0.05842388 -0.5702348-0.05842388 -0.5702348-0.05842388 -0.5702348-0.05842388 -0.5702348-0.058428	A-1986469 35 decides 1987 40 contribution (asked 1985 1987 1988 1
ypswarms.singele.global_best_ 10.48240031 0.82478914 10.48240312 0.82478914 0.482840312 0.82478914 0.482840312 0.82478914 0.482840312 0.82478914 0.482840312 0.82478914 0.482840312 0.82478914 0.482840312 0.82578484 0.482840312 0.82578484 0.482840312 0.8258484 0.482840312 0.8258484 0.282810-830 2826.8484, 820 0.282810-830 2826.848, 820 0.282810-830 2826.848, 820 0.282810-830 2826.848, 820 0.282810-830 2826.848, 820 0.082810-830 -0.82682982 0.08	Compared
ygax_mens_single_global_be yb_bares_single_global_be yb_bares_single_g	Section
## 1908. ## 1909. ##	Commonwealth Comm
900. 1.0000. 1.0000. 1.0000. 1.0000. 1.0000. 1.0000. 1.0000. 1.0000. 1.0	Control (1997) The contro
OSSENDATION CONTRACTOR OSSENDATION CONTR	Compared Conference
The content of the	The company of the co

In []:		<pre>#TESTING print(" TESTING RMSE") print("") start_time=time.time() h_test=np.dot(Test_features,new_a) sum_test=h_test+new_b activation_test=relu(sum_test) y_test=np.dot(activation_test,beta_overall) testing_rmse=RMSE(y_test,Test_Label) print(testing_rmse) end_time=round(time.time()-start_time,3) print("") print("TESTING TIME",end_time)</pre>
	In []: In []:	0.19595105287909553 TESTING TIME 0.002