BOHdataset

BOHdataset = 318×49 table

...

	PublicationNo	Reference	Year	Data	CatalystNameCoding	Au
1	1	"10.1016/j....	2020	1	AuPdCeO2 nanorod	1
2	NaN	""	NaN	2	AuPdCeO2 nanorod	1
3	NaN	""	NaN	3	AuPdCeO2 nanorod	1
4	NaN	""	NaN	4	AuPdCeO2 nanorod	1
5	NaN	""	NaN	5	AuPdCeO2 nanorod	1
6	NaN	""	NaN	6	AuPdCeO2 nanorod	1
7	NaN	""	NaN	7	AuPdCeO2 nanocube	1
8	NaN	""	NaN	8	AuPdCeO2 nanocube	1
9	NaN	""	NaN	9	AuPdCeO2 nanocube	1
10	NaN	""	NaN	10	AuPdCeO2 nanocube	1
11	NaN	""	NaN	11	AuPdCeO2 nanocube	1
12	NaN	""	NaN	12	AuPdCeO2 nanocube	1
13	NaN	""	NaN	13	AuPdCeO2 nanopolyh...	1.0000
14	NaN	""	NaN	14	AuPdCeO2 nanopolyh...	1.0000
15	NaN	""	NaN	15	AuPdCeO2 nanopolyh...	1.0000
16	NaN	""	NaN	16	AuPdCeO2 nanopolyh...	1.0000
17	NaN	""	NaN	17	AuPdCeO2 nanopolyh...	1.0000
18	NaN	""	NaN	18	AuPdCeO2 nanopolyh...	1.0000
19	NaN	""	NaN	19	CeO2 nanorod	0
20	NaN	""	NaN	20	CeO2 nanocube	0
21	NaN	""	NaN	21	CeO2 nanopolyhedra	0
22	2	"10.1016/j....	2020	22	AuCeO2 nanorod	0.8700
23	NaN	""	NaN	23	AuBiCeO2 nanorod	0.7500
24	NaN	""	NaN	24	AuCeO2 nanocube	1.0000
25	NaN	""	NaN	25	AuBiCeO2 nanocube	1.3000
26	3	"10.1016/j....	2019	26	Au/CZ	2.5000
27	NaN	""	NaN	27	Au/CZ	2.5000
28	NaN	""	NaN	28	Au/CZ	2.5000
29	NaN	""	NaN	29	Au/CZ	2.5000
30	NaN	""	NaN	30	0.8AuPdCZO250	2.4000
31	NaN	""	NaN	31	0.8AuPdCZO250	2.4000

	PublicationNo	Reference	Year	Data	CatalystNameCoding	Au
32	NaN	""	NaN	32	0.8AuPdCZO250	2.4000
33	NaN	""	NaN	33	0.8AuPdCZO250	2.4000
34	NaN	""	NaN	34	0.8AuPdCZO450	2.4000
35	NaN	""	NaN	35	0.8AuPdCZO450	2.4000
36	NaN	""	NaN	36	0.8AuPdCZO450	2.4000
37	NaN	""	NaN	37	0.8AuPdCZO700	2.4000
38	NaN	""	NaN	38	0.8AuPdCZO700	2.4000
39	NaN	""	NaN	39	0.8AuPdCZO700	2.4000
40	NaN	""	NaN	40	0.8AuPdCZO700	2.4000
41	NaN	""	NaN	41	Pd/CZ	0
42	NaN	""	NaN	42	Pd/CZ	0
43	NaN	""	NaN	43	Pd/CZ	0
44	NaN	""	NaN	44	Pd/CZ	0
45	4	"10.1002/sl...	2019	45	CeO2-com	0
46	NaN	""	NaN	46	CeO2-cube	0
47	NaN	""	NaN	47	CeO2-rod	0
48	NaN	""	NaN	48	CeO2-poly	0
49	NaN	""	NaN	49	CeO2-meso	0
50	NaN	""	NaN	50	Pd/CeO2-com	0
51	NaN	""	NaN	51	Pd/CeO2-cube	0
52	NaN	""	NaN	52	Pd/CeO2-rod	0
53	NaN	""	NaN	53	Pd/CeO2-poly	0
54	NaN	""	NaN	54	Pd/CeO2-meso	0
55	NaN	""	NaN	55	0.5Pd/CeO2-poly	0
56	NaN	""	NaN	56	1Pd/CeO2-poly	0
57	NaN	""	NaN	57	1.5Pd/CeO2-poly	0
58	NaN	""	NaN	58	Pd/CeO2-poly	0
59	NaN	""	NaN	59	3Pd/CeO2-poly	0
60	NaN	""	NaN	60	Pd/CeO2-poly	0
61	NaN	""	NaN	61	Pd/CeO2-poly	0
62	NaN	""	NaN	62	Pd/CeO2-poly	0
63	NaN	""	NaN	63	Pd/CeO2-poly	0
64	NaN	""	NaN	64	Pd/CeO2-poly	0
65	NaN	""	NaN	65	Pd/CeO2-poly	0

	PublicationNo	Reference	Year	Data	CatalystNameCoding	Au
66	NaN	""	NaN	66	Pd/CeO2-poly	0
67	NaN	""	NaN	67	Pd/CeO2-poly	0
68	NaN	""	NaN	68	Pd/CeO2-poly	0
69	NaN	""	NaN	69	Pd/CeO2-poly	0
70	NaN	""	NaN	70	Pd/CeO2-poly	0
71	5	"10.1021/ac...	2019	71	Au-SA/CeO2-NR	0.9200
72	NaN	""	NaN	72	Au-SA/CeO2-NR	0.9200
73	NaN	""	NaN	73	Au-SA/CeO2-NR	0.9200
74	NaN	""	NaN	74	Au-SA/CeO2-NR	0.9200
75	NaN	""	NaN	75	Au-SA/CeO2-NR	0.9200
76	NaN	""	NaN	76	Au-SA/CeO2-NR	0.9200
77	NaN	""	NaN	77	Au-SA/CeO2-NR	0.9200
78	NaN	""	NaN	78	Au-SA/CeO2-NR	0.9200
79	NaN	""	NaN	79	Au-SA/CeO2-NR	0.9200
80	NaN	""	NaN	80	Au-SA/CeO2-NR	0.9200
81	NaN	""	NaN	81	Au-SA/CeO2-NR	0.9200
82	NaN	""	NaN	82	Au-SA/CeO2-NR	0.9200
83	NaN	""	NaN	83	Au-SA/CeO2-NR	0.9200
84	NaN	""	NaN	84	Au-SA/CeO2-NR	0.9200
85	NaN	""	NaN	85	Au-SA/CeO2-NR	0.9200
86	NaN	""	NaN	86	Au-NC/CeO2-NR	2.3500
87	NaN	""	NaN	87	Au-NC/CeO2-NR	2.3500
88	NaN	""	NaN	88	Au-NC/CeO2-NR	2.3500
89	NaN	""	NaN	89	Au-NC/CeO2-NR	2.3500
90	NaN	""	NaN	90	Au-NC/CeO2-NR	2.3500
91	NaN	""	NaN	91	Au-NC/CeO2-NR	2.3500
92	NaN	""	NaN	92	Au-NC/CeO2-NR	2.3500
93	NaN	""	NaN	93	Au-NC/CeO2-NR	2.3500
94	NaN	""	NaN	94	Au-NC/CeO2-NR	2.3500
95	NaN	""	NaN	95	Au-NC/CeO2-NR	2.3500
96	NaN	""	NaN	96	Au-NC/CeO2-NR	2.3500
97	NaN	""	NaN	97	Au-NC/CeO2-NR	2.3500
98	NaN	""	NaN	98	Au-NC/CeO2-NR	2.3500
99	NaN	""	NaN	99	Au-NC/CeO2-NR	2.3500

	PublicationNo	Reference	Year	Data	CatalystNameCoding	Au
100	NaN	""	NaN	100	Au-NP/CeO2-NR	0.9500

⋮

```
avgParticleAu = nanmean(BOHdataset.pAu);% replacing missing particle sizes with average values
avgParticlePd = nanmean(BOHdataset.pPd);

BOHdataset.pAu(isnan(BOHdataset.pAu)) = avgParticleAu;
BOHdataset.pPd(isnan(BOHdataset.pPd)) = avgParticlePd;
BOHdataset.PrepMethod(ismissing(BOHdataset.PrepMethod)) = "DP";
SurfaceArea = grpstats(BOHdataset(:,{'CeO2Type','SA'}), 'CeO2Type') %Averaging the surface area
```

SurfaceArea = 10×3 table

	CeO2Type	GroupCount	mean_SA
1 commercial	commercial	7	15.2857
2 foam	foam	12	48.4167
3 mesoporous	mesoporous	88	96.7938
4 nanocube	nanocube	14	51.6769
5 nanoparticle	nanoparticle	81	101.2087
6 nanopolyhedra	nanopolyhe...	25	77.0000
7 nanoporous	nanoporous	1	57.1000
8 nanorod	nanorod	85	92.5227
9 nanotube	nanotube	4	81.0000
10 nonporous	nonporous	1	4.7000

```
%disp(SurfaceArea)
BOHdataset = grouptransform(BOHdataset,"CeO2Type","meanfill","SA")
```

BOHdataset = 318×49 table

...

	PublicationNo	Reference	Year	Data	CatalystNameCoding	Au
1	1	"10.1016/j....	2020	1	AuPdCeO2 nanorod	1.0000
2	NaN	""	NaN	2	AuPdCeO2 nanorod	1.0000
3	NaN	""	NaN	3	AuPdCeO2 nanorod	1.0000
4	NaN	""	NaN	4	AuPdCeO2 nanorod	1.0000
5	NaN	""	NaN	5	AuPdCeO2 nanorod	1.0000
6	NaN	""	NaN	6	AuPdCeO2 nanorod	1.0000
7	NaN	""	NaN	7	AuPdCeO2 nanocube	1.0000
8	NaN	""	NaN	8	AuPdCeO2 nanocube	1.0000

	PublicationNo	Reference	Year	Data	CatalystNameCoding	Au
9	NaN	""	NaN	9	AuPdCeO2 nanocube	1.0000
10	NaN	""	NaN	10	AuPdCeO2 nanocube	1.0000
11	NaN	""	NaN	11	AuPdCeO2 nanocube	1.0000
12	NaN	""	NaN	12	AuPdCeO2 nanocube	1.0000
13	NaN	""	NaN	13	AuPdCeO2 nanopolyh...	1.0000
14	NaN	""	NaN	14	AuPdCeO2 nanopolyh...	1.0000
15	NaN	""	NaN	15	AuPdCeO2 nanopolyh...	1.0000
16	NaN	""	NaN	16	AuPdCeO2 nanopolyh...	1.0000
17	NaN	""	NaN	17	AuPdCeO2 nanopolyh...	1.0000
18	NaN	""	NaN	18	AuPdCeO2 nanopolyh...	1.0000
19	NaN	""	NaN	19	CeO2 nanorod	0
20	NaN	""	NaN	20	CeO2 nanocube	0
21	NaN	""	NaN	21	CeO2 nanopolyhedra	0
22	2	"10.1016/j....	2020	22	AuCeO2 nanorod	0.8700
23	NaN	""	NaN	23	AuBiCeO2 nanorod	0.7500
24	NaN	""	NaN	24	AuCeO2 nanocube	1.0000
25	NaN	""	NaN	25	AuBiCeO2 nanocube	1.3000
26	3	"10.1016/j....	2019	26	Au/CZ	2.5000
27	NaN	""	NaN	27	Au/CZ	2.5000
28	NaN	""	NaN	28	Au/CZ	2.5000
29	NaN	""	NaN	29	Au/CZ	2.5000
30	NaN	""	NaN	30	0.8AuPdCZO250	2.4000
31	NaN	""	NaN	31	0.8AuPdCZO250	2.4000
32	NaN	""	NaN	32	0.8AuPdCZO250	2.4000
33	NaN	""	NaN	33	0.8AuPdCZO250	2.4000
34	NaN	""	NaN	34	0.8AuPdCZO450	2.4000
35	NaN	""	NaN	35	0.8AuPdCZO450	2.4000
36	NaN	""	NaN	36	0.8AuPdCZO450	2.4000
37	NaN	""	NaN	37	0.8AuPdCZO700	2.4000
38	NaN	""	NaN	38	0.8AuPdCZO700	2.4000
39	NaN	""	NaN	39	0.8AuPdCZO700	2.4000
40	NaN	""	NaN	40	0.8AuPdCZO700	2.4000
41	NaN	""	NaN	41	Pd/CZ	0
42	NaN	""	NaN	42	Pd/CZ	0

	PublicationNo	Reference	Year	Data	CatalystNameCoding	Au
43	NaN	""	NaN	43	Pd/CZ	0
44	NaN	""	NaN	44	Pd/CZ	0
45	4	"10.1002/sl...	2019	45	CeO2-com	0
46	NaN	""	NaN	46	CeO2-cube	0
47	NaN	""	NaN	47	CeO2-rod	0
48	NaN	""	NaN	48	CeO2-poly	0
49	NaN	""	NaN	49	CeO2-meso	0
50	NaN	""	NaN	50	Pd/CeO2-com	0
51	NaN	""	NaN	51	Pd/CeO2-cube	0
52	NaN	""	NaN	52	Pd/CeO2-rod	0
53	NaN	""	NaN	53	Pd/CeO2-poly	0
54	NaN	""	NaN	54	Pd/CeO2-meso	0
55	NaN	""	NaN	55	0.5Pd/CeO2-poly	0
56	NaN	""	NaN	56	1Pd/CeO2-poly	0
57	NaN	""	NaN	57	1.5Pd/CeO2-poly	0
58	NaN	""	NaN	58	Pd/CeO2-poly	0
59	NaN	""	NaN	59	3Pd/CeO2-poly	0
60	NaN	""	NaN	60	Pd/CeO2-poly	0
61	NaN	""	NaN	61	Pd/CeO2-poly	0
62	NaN	""	NaN	62	Pd/CeO2-poly	0
63	NaN	""	NaN	63	Pd/CeO2-poly	0
64	NaN	""	NaN	64	Pd/CeO2-poly	0
65	NaN	""	NaN	65	Pd/CeO2-poly	0
66	NaN	""	NaN	66	Pd/CeO2-poly	0
67	NaN	""	NaN	67	Pd/CeO2-poly	0
68	NaN	""	NaN	68	Pd/CeO2-poly	0
69	NaN	""	NaN	69	Pd/CeO2-poly	0
70	NaN	""	NaN	70	Pd/CeO2-poly	0
71	5	"10.1021/ac...	2019	71	Au-SA/CeO2-NR	0.9200
72	NaN	""	NaN	72	Au-SA/CeO2-NR	0.9200
73	NaN	""	NaN	73	Au-SA/CeO2-NR	0.9200
74	NaN	""	NaN	74	Au-SA/CeO2-NR	0.9200
75	NaN	""	NaN	75	Au-SA/CeO2-NR	0.9200
76	NaN	""	NaN	76	Au-SA/CeO2-NR	0.9200

	PublicationNo	Reference	Year	Data	CatalystNameCoding	Au
77	NaN	""	NaN	77	Au-SA/CeO2-NR	0.9200
78	NaN	""	NaN	78	Au-SA/CeO2-NR	0.9200
79	NaN	""	NaN	79	Au-SA/CeO2-NR	0.9200
80	NaN	""	NaN	80	Au-SA/CeO2-NR	0.9200
81	NaN	""	NaN	81	Au-SA/CeO2-NR	0.9200
82	NaN	""	NaN	82	Au-SA/CeO2-NR	0.9200
83	NaN	""	NaN	83	Au-SA/CeO2-NR	0.9200
84	NaN	""	NaN	84	Au-SA/CeO2-NR	0.9200
85	NaN	""	NaN	85	Au-SA/CeO2-NR	0.9200
86	NaN	""	NaN	86	Au-NC/CeO2-NR	2.3500
87	NaN	""	NaN	87	Au-NC/CeO2-NR	2.3500
88	NaN	""	NaN	88	Au-NC/CeO2-NR	2.3500
89	NaN	""	NaN	89	Au-NC/CeO2-NR	2.3500
90	NaN	""	NaN	90	Au-NC/CeO2-NR	2.3500
91	NaN	""	NaN	91	Au-NC/CeO2-NR	2.3500
92	NaN	""	NaN	92	Au-NC/CeO2-NR	2.3500
93	NaN	""	NaN	93	Au-NC/CeO2-NR	2.3500
94	NaN	""	NaN	94	Au-NC/CeO2-NR	2.3500
95	NaN	""	NaN	95	Au-NC/CeO2-NR	2.3500
96	NaN	""	NaN	96	Au-NC/CeO2-NR	2.3500
97	NaN	""	NaN	97	Au-NC/CeO2-NR	2.3500
98	NaN	""	NaN	98	Au-NC/CeO2-NR	2.3500
99	NaN	""	NaN	99	Au-NC/CeO2-NR	2.3500
100	NaN	""	NaN	100	Au-NP/CeO2-NR	0.9500

⋮

```
BOHdataset.SA
```

```
ans = 318×1
   107
   107
   107
   107
   107
   107
    49
    49
    49
    49
```

```
%for i = 1:height(SurfaceArea) %i.e for each class
   % BOHdataset.SA(BOHdataset.CeO2Type == i & isnan(BOHdataset.SA)) = SurfaceArea.mean_SA(i);

%end
DPrepMethod = dummyvar(BOHdataset.PrepMethod);
DCeO2Type = dummyvar(BOHdataset.CeO2Type);
%DCeO2PrepMethod = dummyvar(BOHdataset.CeO2PrepMethod);
DSolvent = dummyvar(BOHdataset.Solvent);
%DPlanes = dummyvar(BOHdataset.ActivePlane);

%DCeO2PrepMethod = array2table(DCeO2PrepMethod);
DCeO2Type = array2table(DCeO2Type);
DPrepMethod = array2table(DPrepMethod);
DSolvent = array2table(DSolvent);



BOHdataset1 = [BOHdataset, DCeO2Type, DPrepMethod, DSolvent]
```

BOHdataset1 = 318×79 table

...

	PublicationNo	Reference	Year	Data	CatalystNameCoding	Au
1	1	"10.1016/j....	2020	1	AuPdCeO2 nanorod	1.0000
2	NaN	""	NaN	2	AuPdCeO2 nanorod	1.0000
3	NaN	""	NaN	3	AuPdCeO2 nanorod	1.0000
4	NaN	""	NaN	4	AuPdCeO2 nanorod	1.0000
5	NaN	""	NaN	5	AuPdCeO2 nanorod	1.0000
6	NaN	""	NaN	6	AuPdCeO2 nanorod	1.0000
7	NaN	""	NaN	7	AuPdCeO2 nanocube	1.0000
8	NaN	""	NaN	8	AuPdCeO2 nanocube	1.0000
9	NaN	""	NaN	9	AuPdCeO2 nanocube	1.0000
10	NaN	""	NaN	10	AuPdCeO2 nanocube	1.0000
11	NaN	""	NaN	11	AuPdCeO2 nanocube	1.0000
12	NaN	""	NaN	12	AuPdCeO2 nanocube	1.0000
13	NaN	""	NaN	13	AuPdCeO2 nanopolyh...	1.0000
14	NaN	""	NaN	14	AuPdCeO2 nanopolyh...	1.0000
15	NaN	""	NaN	15	AuPdCeO2 nanopolyh...	1.0000
16	NaN	""	NaN	16	AuPdCeO2 nanopolyh...	1.0000
17	NaN	""	NaN	17	AuPdCeO2 nanopolyh...	1.0000

	PublicationNo	Reference	Year	Data	CatalystNameCoding	Au
18	NaN	""	NaN	18	AuPdCeO2 nanopolyh...	1.0000
19	NaN	""	NaN	19	CeO2 nanorod	0
20	NaN	""	NaN	20	CeO2 nanocube	0
21	NaN	""	NaN	21	CeO2 nanopolyhedra	0
22	2	"10.1016/j....	2020	22	AuCeO2 nanorod	0.8700
23	NaN	""	NaN	23	AuBiCeO2 nanorod	0.7500
24	NaN	""	NaN	24	AuCeO2 nanocube	1.0000
25	NaN	""	NaN	25	AuBiCeO2 nanocube	1.3000
26	3	"10.1016/j....	2019	26	Au/CZ	2.5000
27	NaN	""	NaN	27	Au/CZ	2.5000
28	NaN	""	NaN	28	Au/CZ	2.5000
29	NaN	""	NaN	29	Au/CZ	2.5000
30	NaN	""	NaN	30	0.8AuPdCZO250	2.4000
31	NaN	""	NaN	31	0.8AuPdCZO250	2.4000
32	NaN	""	NaN	32	0.8AuPdCZO250	2.4000
33	NaN	""	NaN	33	0.8AuPdCZO250	2.4000
34	NaN	""	NaN	34	0.8AuPdCZO450	2.4000
35	NaN	""	NaN	35	0.8AuPdCZO450	2.4000
36	NaN	""	NaN	36	0.8AuPdCZO450	2.4000
37	NaN	""	NaN	37	0.8AuPdCZO700	2.4000
38	NaN	""	NaN	38	0.8AuPdCZO700	2.4000
39	NaN	""	NaN	39	0.8AuPdCZO700	2.4000
40	NaN	""	NaN	40	0.8AuPdCZO700	2.4000
41	NaN	""	NaN	41	Pd/CZ	0
42	NaN	""	NaN	42	Pd/CZ	0
43	NaN	""	NaN	43	Pd/CZ	0
44	NaN	""	NaN	44	Pd/CZ	0
45	4	"10.1002/sl...	2019	45	CeO2-com	0
46	NaN	""	NaN	46	CeO2-cube	0
47	NaN	""	NaN	47	CeO2-rod	0
48	NaN	""	NaN	48	CeO2-poly	0
49	NaN	""	NaN	49	CeO2-meso	0
50	NaN	""	NaN	50	Pd/CeO2-com	0
51	NaN	""	NaN	51	Pd/CeO2-cube	0

	PublicationNo	Reference	Year	Data	CatalystNameCoding	Au
52	NaN	""	NaN	52	Pd/CeO2-rod	0
53	NaN	""	NaN	53	Pd/CeO2-poly	0
54	NaN	""	NaN	54	Pd/CeO2-meso	0
55	NaN	""	NaN	55	0.5Pd/CeO2-poly	0
56	NaN	""	NaN	56	1Pd/CeO2-poly	0
57	NaN	""	NaN	57	1.5Pd/CeO2-poly	0
58	NaN	""	NaN	58	Pd/CeO2-poly	0
59	NaN	""	NaN	59	3Pd/CeO2-poly	0
60	NaN	""	NaN	60	Pd/CeO2-poly	0
61	NaN	""	NaN	61	Pd/CeO2-poly	0
62	NaN	""	NaN	62	Pd/CeO2-poly	0
63	NaN	""	NaN	63	Pd/CeO2-poly	0
64	NaN	""	NaN	64	Pd/CeO2-poly	0
65	NaN	""	NaN	65	Pd/CeO2-poly	0
66	NaN	""	NaN	66	Pd/CeO2-poly	0
67	NaN	""	NaN	67	Pd/CeO2-poly	0
68	NaN	""	NaN	68	Pd/CeO2-poly	0
69	NaN	""	NaN	69	Pd/CeO2-poly	0
70	NaN	""	NaN	70	Pd/CeO2-poly	0
71	5	"10.1021/ac...	2019	71	Au-SA/CeO2-NR	0.9200
72	NaN	""	NaN	72	Au-SA/CeO2-NR	0.9200
73	NaN	""	NaN	73	Au-SA/CeO2-NR	0.9200
74	NaN	""	NaN	74	Au-SA/CeO2-NR	0.9200
75	NaN	""	NaN	75	Au-SA/CeO2-NR	0.9200
76	NaN	""	NaN	76	Au-SA/CeO2-NR	0.9200
77	NaN	""	NaN	77	Au-SA/CeO2-NR	0.9200
78	NaN	""	NaN	78	Au-SA/CeO2-NR	0.9200
79	NaN	""	NaN	79	Au-SA/CeO2-NR	0.9200
80	NaN	""	NaN	80	Au-SA/CeO2-NR	0.9200
81	NaN	""	NaN	81	Au-SA/CeO2-NR	0.9200
82	NaN	""	NaN	82	Au-SA/CeO2-NR	0.9200
83	NaN	""	NaN	83	Au-SA/CeO2-NR	0.9200
84	NaN	""	NaN	84	Au-SA/CeO2-NR	0.9200
85	NaN	""	NaN	85	Au-SA/CeO2-NR	0.9200

	PublicationNo	Reference	Year	Data	CatalystNameCoding	Au
86	NaN	""	NaN	86	Au-NC/CeO2-NR	2.3500
87	NaN	""	NaN	87	Au-NC/CeO2-NR	2.3500
88	NaN	""	NaN	88	Au-NC/CeO2-NR	2.3500
89	NaN	""	NaN	89	Au-NC/CeO2-NR	2.3500
90	NaN	""	NaN	90	Au-NC/CeO2-NR	2.3500
91	NaN	""	NaN	91	Au-NC/CeO2-NR	2.3500
92	NaN	""	NaN	92	Au-NC/CeO2-NR	2.3500
93	NaN	""	NaN	93	Au-NC/CeO2-NR	2.3500
94	NaN	""	NaN	94	Au-NC/CeO2-NR	2.3500
95	NaN	""	NaN	95	Au-NC/CeO2-NR	2.3500
96	NaN	""	NaN	96	Au-NC/CeO2-NR	2.3500
97	NaN	""	NaN	97	Au-NC/CeO2-NR	2.3500
98	NaN	""	NaN	98	Au-NC/CeO2-NR	2.3500
99	NaN	""	NaN	99	Au-NC/CeO2-NR	2.3500
100	NaN	""	NaN	100	Au-NP/CeO2-NR	0.9500

⋮

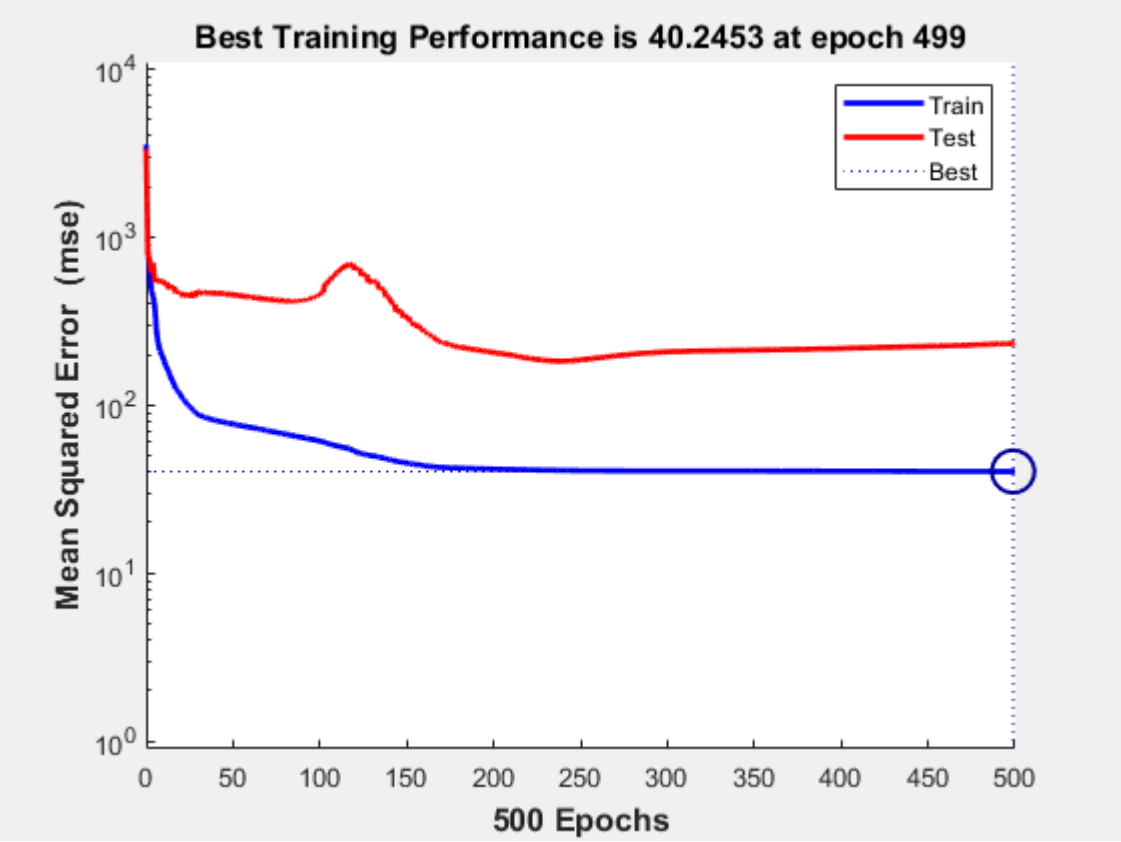
```
BOHdataset1.PrepMethod = [];
BOHdataset1.CeO2Type = [];
BOHdataset1.CeO2PrepMethod = [];
BOHdataset1.Solvent = [];



inputs1 = removevars(BOHdataset1,  {'Data','PublicationNo', 'VarName48','Reference','Remarks','
% Active plane, Crystallite size, pore radius and pore volume are removed
% due to too many missing values

inputs = double(inputs1{:,:});

InputsFilled = fillmissing(inputs, "constant", 0);

InputsandTarget = [InputsFilled, BOHdataset1.Conversion];
targets = BOHdataset1.Conversion;
transposedInput = transpose(InputsFilled);
min(InputsFilled);
max(InputsFilled);
%maximum-minimum normalization performed on the data
```

```
 % Solve an Input-Output Fitting problem with a Neural Network
% Script generated by Neural Fitting app

% This script assumes these variables are defined:
%
%   Input_train - input data.
%   Target_train - target data.
x = transposedInput;
%xpAu = transposedInputpAu;
t = transposedTarget;
```

```
rng('default') ;%random number generator set to default
s = rng;
 %nntool
% Choose a Training Function
% For a list of all training functions type: help nntrain
% 'trainlm' is usually fastest.
% 'trainbr' takes longer but may be better for challenging problems.
% 'trainscg' uses less memory. Suitable in low memory situations.
trainFcn = 'trainbr';  % Levenberg-Marquardt backpropagation.
% Create a Fitting Network
h1 = 20; %more hidden layers
h2 = 20;

 net = fitnet([h1, h2],trainFcn);
 net.trainParam.epochs= 500; %more epochs
% Choose Input and Output Pre/Post-Processing Functions
% For a list of all processing functions type: help nnprocess
net.input.processFcns = {'removeconstantrows','mapminmax'};
net.output.processFcns = {'removeconstantrows','mapminmax'};
% Setup Division of Data for Training, Validation, Testing
% For a list of all data division functions type: help nndivide
net.divideFcn = 'dividerand';  % Divide data randomly
net.divideMode = 'sample';  % Divide up every sample
net.divideParam.trainRatio = 80/100;
net.divideParam.valRatio = 10/100;
net.divideParam.testRatio = 10/100;
% Choose a Performance Function
% For a list of all performance functions type: help nnperformance
net.performFcn = 'mse';  % Mean Squared Error
% Choose Plot Functions
% For a list of all plot functions type: help nnplot
```

```
net.plotFcns = {'plotperform','plottrainstate','ploterrhist', ...
  'plotregression', 'plotfit'};
```

```
rng('default')
% Train the Network
[net,tr] = train(net,x,t);
```




```
%[netpAu, trpAu] = train(net, xpAu,t);

% Test the Network
y = net(x);
e = gsubtract(t,y);
performance = perform(net,t,y)
```

```
performance = 59.5659
```

```
totalRMSE = sqrt(performance)
```

```
totalRMSE = 7.7179
```

```
%ypAu = netpAu(xpAu);
%e = gsubtract(t,ypAu);
%performance = perform(netpAu,t,ypAu)

% Recalculate Training, Validation and Test Performance
trainTargets = t .* tr.trainMask{1};
valTargets = t .* tr.valMask{1};
testTargets = t .* tr.testMask{1};
```

```
trainPerformance = perform(net,trainTargets,y)
```

```
trainPerformance = 40.2453
```

```
%trainPerformancepAu = perform(net, trainTargets, ypAu)
trainRMSE = sqrt(trainPerformance)
```

```
trainRMSE = 6.3439
```

```
%trainRMSEpAu = sqrt(trainPerformancepAu);
valPerformance = perform(net,valTargets,y)
```

```
valPerformance = NaN
```

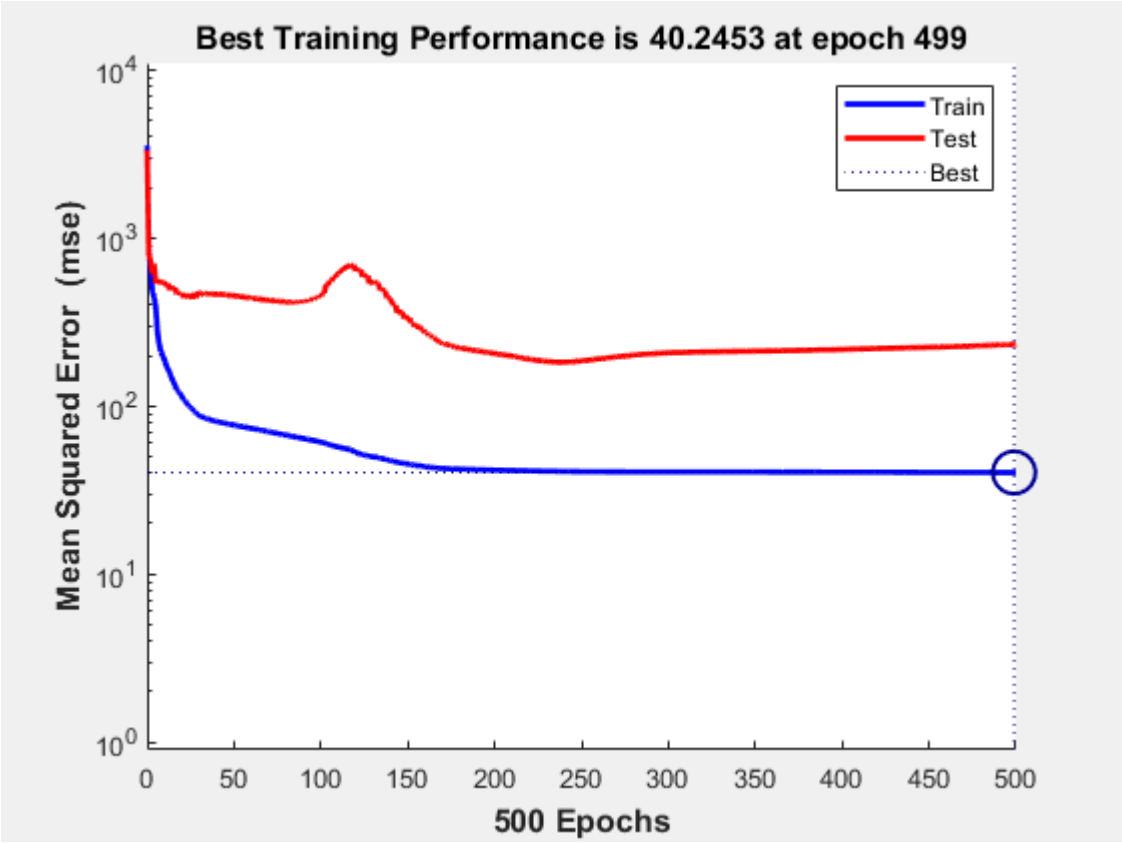
```
testPerformance = perform(net,testTargets,y)
```

```
testPerformance = 232.2431
```

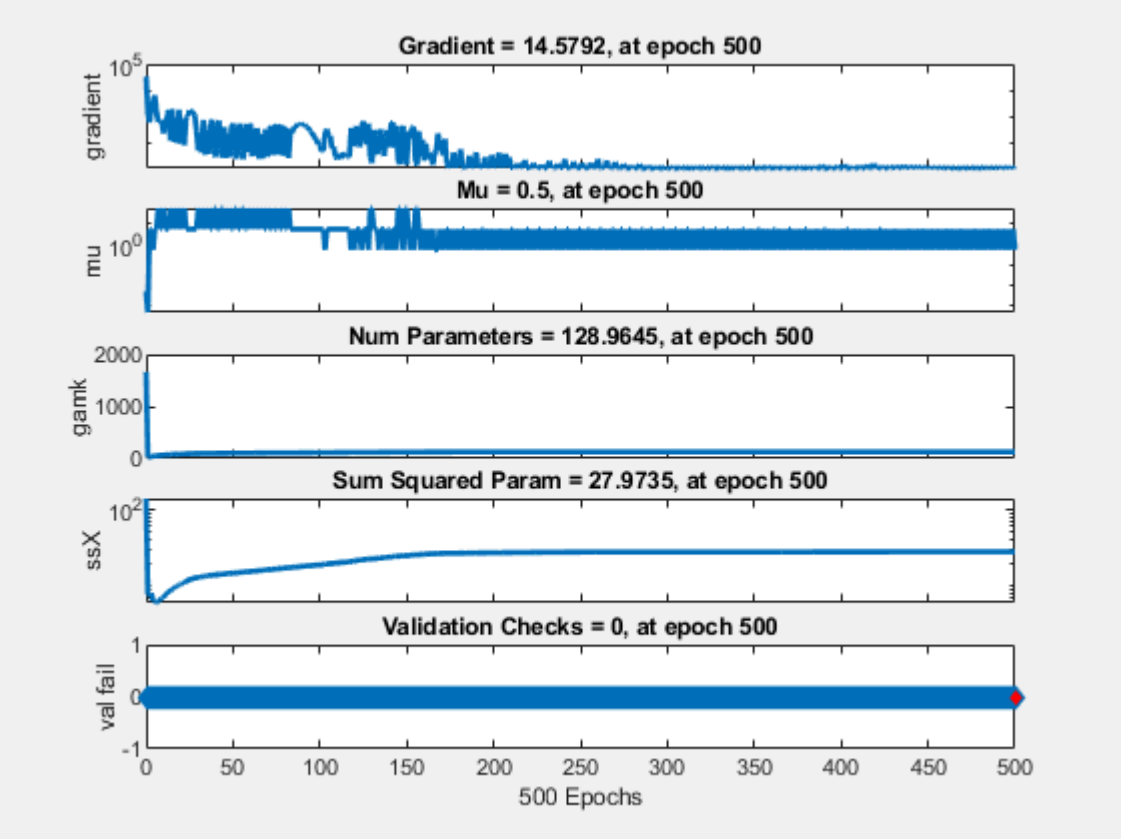
```
%testPerformancepAu = perform(net,testTargets,ypAu)

testRMSE = sqrt(testPerformance)
```

```
testRMSE = 15.2395
```

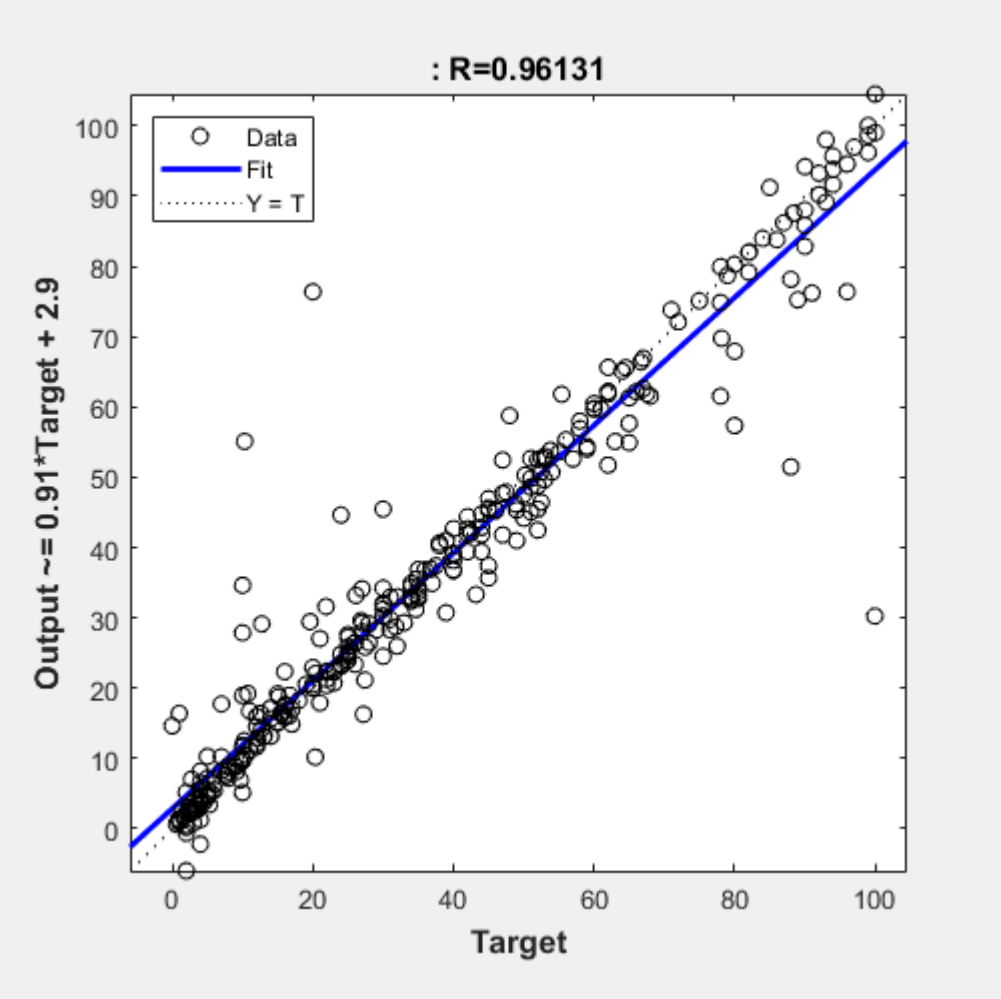
```
%testRMSEpAu = sqrt(testPerformancepAu)
% View the Network
view(net)
% Plots
% Uncomment these lines to enable various plots.
figure, plotperform(tr)
```



```
figure, plottrainstate(tr)
```




```
figure, ploterrhist(e)
figure, plotregression(t,y)
```



Partial Derivative of the Output Data

```
dydx = gradient(y, BOHdataset1.TemperatureK)
```

```
dydx = 1×318
       Inf       Inf       Inf       Inf       Inf      -Inf      -Inf       Inf ···
```

Save Network Weights and Biasses

```
BOHnet = net;
save BOHnet;
IW = net.IW; %weights for the input to hidden layer
LW = net.LW %weights for hidden layer to output
```

```
LW = 3×3 cell array
    { 0×0  double}    {0×0  double}    {0×0 double}
    {20×20 double}    {0×0  double}    {0×0 double}
    { 0×0  double}    {1×20 double}    {0×0 double}
```

```
b = net.b %bias values
```

```
b = 3×1 cell array
    {20×1 double}
    {20×1 double}
```

```
{[  -0.0835]}
```

```
save IW
save LW
save b
```

```

```

Variable Type: Noble Metal

```
inputNM = inputs1
```

```
inputNM = 318×63 table
```

...

	Au	Pd	pAu	pPd	CalcTK	CalcThr	CeO2
1	1	0.5400	3.0500	3.0500	573	2	1
2	1	0.5400	3.0500	3.0500	573	2	1
3	1	0.5400	3.0500	3.0500	573	2	1
4	1	0.5400	3.0500	3.0500	573	2	1
5	1	0.5400	3.0500	3.0500	573	2	1
6	1	0.5400	3.0500	3.0500	573	2	1
7	1	0.5400	2.9200	2.9200	573	2	1
8	1	0.5400	2.9200	2.9200	573	2	1
9	1	0.5400	2.9200	2.9200	573	2	1
10	1	0.5400	2.9200	2.9200	573	2	1

⋮

```
inputNM.Au = zeros([318 1]); inputNM.Pd = zeros([318 1]); inputNM.pAu = zeros([318 1]); inputNM
inputNM = double(inputNM{:,:});

inputNM = fillmissing(inputNM, "constant", 0);
transposedInputNM = transpose(inputNM);
xNM = transposedInputNM;
rng('default');
[net,tr] = train(net,xNM,t);
```

```
Undefined function or variable 'net'.
```

```

yNM = net(xNM);
e = gsubtract(t,yNM);
performance = perform(net,t,yNM)
totalRMSENM = sqrt(performance)
RImpNM = (totalRMSENM - totalRMSE)
% Recalculate Training, Validation and Test Performance
```

```
trainTargets = t .* tr.trainMask{1};
valTargets = t .* tr.valMask{1};
testTargets = t .* tr.testMask{1};
trainPerformance = perform(net,trainTargets,yNM)
trainRMSE = sqrt(trainPerformance)
testPerformance = perform(net,testTargets,yNM)
testRMSE = sqrt(testPerformance)
```

```
yTrain = net(x(:, tr.trainInd));
yTrainTrue = y(tr.trainInd);
```

```
dydx1 = gradient(network1_outputs, BOHdataset1.ReactionTimeh)
plot(BOHdataset1.ReactionTimeh, dydx1)
```