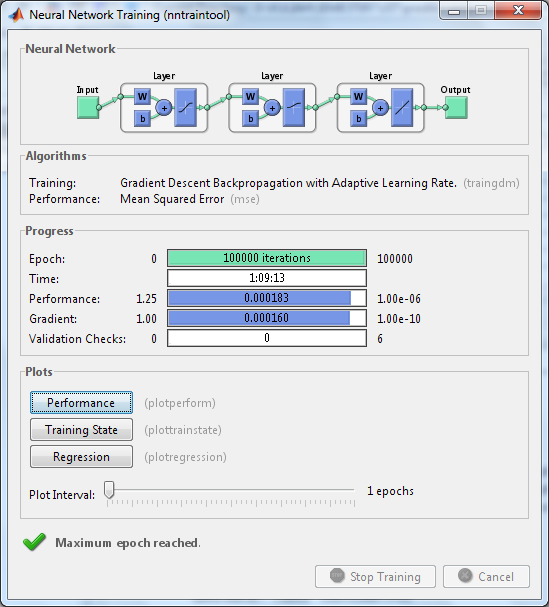
**Tugas JST | Peramalan Jumlah Penjualan**

**Rischan Mafrur / 09650007**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Jumlah data | epochs | goal | lr | mc |
| 50 | 100000 | 1e-6 | 0.5 | 0.8 |

Training berlangsung selama 1 jam 9 menit 13 detik menggunakan intel Atom N570. Training Selesai sebelum goal tercapai yaitu setelah epochs terpenuhi 100000 epochs.



Testing menggunakan data training => semua data dikenali / 100 % data dikenali

Hasil testing:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| t | 16 | 13 | **15** | 28 | 30 | 12 | 13 | 14 | 15 | 13 | 35 | 38 | **11** | 12 |
| hasil | 16,7 | 17,3 | **14,9** | 29,7 | 28,3 | 10,7 | 12,2 | 10,9 | 12,1 | 4,6 | 25,6 | 26,6 | **11,3** | 16,9 |

Hasil testing menggunakan data testing ada 2 data yang dikenali dan 12 data yang tidak dikenali dari 14 data testing, bisa disimpulkan bahwa testing dapat mengenali **14,3 %** dari keseluruhan data testing.

**Lampiran**

[pn,meanp,stdp,tn,meant,stdt]=prestd(p,t)

net=newff(minmax(pn),[10 5 1],{'tansig' 'logsig' 'purelin'},'traingdm');

net.IW{1,1}=[...

0.4500 0.0894 -0.8542 0.8416

0.9579 -0.1978 -0.5595 -0.6146

-0.6119 -0.9953 0.1435 -0.4015

-0.6320 -0.7867 -0.4583 0.5986

-0.0152 1.1257 0.4672 -0.1576

0.4021 -0.7500 -0.3316 -0.8762

0.5958 0.6346 0.6376 0.6659

-0.3116 0.9297 -0.7423 0.1878

-0.0043 0.7410 -0.8049 -0.6237

-0.7518 0.4902 0.5800 -0.6967];

net.b{1,1}=[...

-2.9086

-1.6140

2.9221

1.9021

-0.9166

1.5395

-1.2132

-1.4846

-1.4403

-2.2184];

net.LW{2,1}=[...

1.4500 1.0501 0.9679 -0.0225 0.0705 -1.7650 0.8302 1.1893 1.1413 0.3847

0.7305 1.4203 -0.3329 0.1057 -0.1429 -1.5011 -1.5074 1.0495 -1.1499 -1.0838

1.5362 -1.2236 -0.6011 0.0316 0.8532 0.4183 -1.4828 -0.9369 -0.9060 1.3406

1.4153 -1.2922 -0.1157 -1.2216 1.0480 1.6869 0.0372 0.9609 -0.1592 -0.8628

0.9604 1.1946 -1.2088 1.4801 0.3484 -0.2468 1.2570 -0.9697 -1.4478 -0.1297];

net.b{2,1}

net.LW{3,2}=[...

0.6595 -0.3997 0.4192 -0.8508 0.5004];

net.b{3,1}=[-0.0622];

net.trainParam.epochs=100000;

net.trainParam.goal=1e-6;

net.trainParam.lr=0.5;

net.trainParam.show=10000;

net.trainParam.mc=0.8;

pn =

Columns 1 through 6

-0.5461 -0.9630 -0.1292 -0.5461 -0.1292 -0.7546

-0.7921 -0.5931 -0.3941 -0.5931 -0.1950 -0.5931

-0.6029 -0.4191 -0.2353 -0.4191 -0.2353 -0.7868

-0.8047 -0.8047 -0.2485 -0.0630 -0.2485 -0.8047

Columns 7 through 12

-0.3377 -0.5461 -0.5461 -0.3377 -0.1292 -0.3377

-0.1950 -0.3941 -0.7921 -0.7921 -0.1950 0.0040

-0.2353 -0.0515 -0.7868 -0.6029 -0.2353 -0.4191

-0.2485 -0.4339 -0.4339 -0.0630 -0.2485 -0.6193

Columns 13 through 18

-0.1292 -0.7546 -0.7546 0.0792 -0.1292 -0.7546

-0.1950 -0.7921 -0.5931 -0.3941 -0.1950 -0.3941

-0.2353 -0.4191 -0.0515 -0.6029 -0.7868 -0.6029

-0.8047 -0.6193 -0.2485 -0.4339 -0.2485 -0.6193

Columns 19 through 24

-0.1292 -0.3377 -0.1292 -0.3377 0.0792 -0.1292

-0.1950 -0.5931 -0.7921 -0.5931 -0.1950 -0.1950

-0.2353 -0.4191 -0.2353 -0.6029 -1.1544 0.3162

0.3078 -0.0630 0.1224 -0.4339 -0.6193 0.3078

Columns 25 through 30

-0.5461 2.3721 2.7890 -0.5461 -0.1292 -0.3377

-0.3941 1.9943 2.3923 -0.5931 0.4020 0.0040

-0.0515 1.7868 2.5221 -0.4191 0.3162 -0.2353

-0.2485 2.7183 3.4599 -0.6193 0.1224 -0.6193

Columns 31 through 36

-0.1292 2.1636 2.5805 -0.5461 0.4961 0.0792

0.2030 1.9943 2.7904 -0.3941 0.4020 -0.1950

-0.9706 2.7059 3.4412 -0.6029 0.1324 -0.6029

-0.0630 2.1620 2.3474 -0.6193 -0.4339 -0.4339

Columns 37 through 42

2.1636 2.9974 -0.3377 0.4961 -0.1292 -0.9630

1.9943 3.7855 -0.5931 0.2030 -0.5931 0.0040

2.1544 2.3382 -0.6029 -0.4191 -0.4191 0.5000

2.3474 2.1620 -0.6193 -0.2485 -0.0630 -0.2485

Columns 43 through 48

-0.5461 0.2877 -0.5461 -0.5461 -0.3377 -0.3377

-0.5931 -0.3941 -0.3941 -0.5931 -0.1950 0.0040

-0.6029 -0.2353 -0.0515 -0.4191 -0.2353 0.3162

-0.4339 -0.2485 0.3078 -0.2485 -0.6193 -0.8047

Columns 49 through 50

-0.5461 -1.1714

-0.3941 -0.1950

-0.2353 -0.0515

-0.4339 -0.2485

meanp =

12.6200

12.9800

13.2800

13.3400

stdp =

4.7975

5.0244

5.4400

5.3932

tn =

Columns 1 through 6

-0.8221 -0.6245 -0.2292 -0.4269 -0.2292 -0.4269

Columns 7 through 12

-0.2292 -0.6245 -0.6245 -0.2292 -0.2292 -0.4269

Columns 13 through 18

-0.2292 -0.6245 -1.2173 -0.0316 0.1660 -0.4269

Columns 19 through 24

0.3636 -0.2292 -1.0197 -0.6245 -0.0316 0.1660

Columns 25 through 30

-0.6245 2.3398 2.5374 -0.6245 -0.4269 -0.4269

Columns 31 through 36

0.5612 2.5374 2.3398 -0.6245 -0.2292 -0.0316

Columns 37 through 42

2.5374 2.7350 -0.4269 -0.2292 0.3636 0.1660

Columns 43 through 48

-0.2292 -0.6245 -0.8221 -0.4269 -0.0316 -0.0316

Columns 49 through 50

-0.4269 -1.0197

meant =

13.1600

stdt =

5.0603

>> net

net =

Neural Network object:

architecture:

numInputs: 1

numLayers: 3

biasConnect: [1; 1; 1]

inputConnect: [1; 0; 0]

layerConnect: [0 0 0; 1 0 0; 0 1 0]

outputConnect: [0 0 1]

numOutputs: 1 (read-only)

numInputDelays: 0 (read-only)

numLayerDelays: 0 (read-only)

subobject structures:

inputs: {1x1 cell} of inputs

layers: {3x1 cell} of layers

outputs: {1x3 cell} containing 1 output

biases: {3x1 cell} containing 3 biases

inputWeights: {3x1 cell} containing 1 input weight

layerWeights: {3x3 cell} containing 2 layer weights

functions:

adaptFcn: 'trains'

divideFcn: (none)

gradientFcn: 'calcgrad'

initFcn: 'initlay'

performFcn: 'mse'

plotFcns: {'plotperform','plottrainstate','plotregression'}

trainFcn: 'traingdm'

parameters:

adaptParam: .passes

divideParam: (none)

gradientParam: (none)

initParam: (none)

performParam: (none)

trainParam: .show, .showWindow, .showCommandLine, .epochs,

.time, .goal, .max\_fail, .lr,

.mc, .min\_grad

weight and bias values:

IW: {3x1 cell} containing 1 input weight matrix

LW: {3x3 cell} containing 2 layer weight matrices

b: {3x1 cell} containing 3 bias vectors

other:

name: ''

userdata: (user information)

>> net.IW{1,1}=[...

0.4500 0.0894 -0.8542 0.8416

0.9579 -0.1978 -0.5595 -0.6146

-0.6119 -0.9953 0.1435 -0.4015

-0.6320 -0.7867 -0.4583 0.5986

-0.0152 1.1257 0.4672 -0.1576

0.4021 -0.7500 -0.3316 -0.8762

0.5958 0.6346 0.6376 0.6659

-0.3116 0.9297 -0.7423 0.1878

-0.0043 0.7410 -0.8049 -0.6237

-0.7518 0.4902 0.5800 -0.6967];

>> net.b{1,1}=[...

-2.9086

-1.6140

2.9221

1.9021

-0.9166

1.5395

-1.2132

-1.4846

-1.4403

-2.2184];

>> net.LW{2,1}=[...

1.4500 1.0501 0.9679 -0.0225 0.0705 -1.7650 0.8302 1.1893 1.1413 0.3847

0.7305 1.4203 -0.3329 0.1057 -0.1429 -1.5011 -1.5074 1.0495 -1.1499 -1.0838

1.5362 -1.2236 -0.6011 0.0316 0.8532 0.4183 -1.4828 -0.9369 -0.9060 1.3406

1.4153 -1.2922 -0.1157 -1.2216 1.0480 1.6869 0.0372 0.9609 -0.1592 -0.8628

0.9604 1.1946 -1.2088 1.4801 0.3484 -0.2468 1.2570 -0.9697 -1.4478 -0.1297];

>> net.b{2,1}

net.LW{3,2}=[...

0.6595 -0.3997 0.4192 -0.8508 0.5004];

net.b{3,1}=[-0.0622];

ans =

3.2889

1.6445

0

1.6445

-3.2889

>> net.trainParam.epochs=100000;

net.trainParam.goal=1e-6;

net.trainParam.lr=0.5;

net.trainParam.show=200;

net.trainParam.mc=0.8;

>> net.trainParam.epochs=100000;

net.trainParam.goal=1e-6;

net.trainParam.lr=0.5;

net.trainParam.show=10000;

net.trainParam.mc=0.8;

>> net.trainParam.mc

ans =

0.8000

>> net.trainParam.show

ans =

10000

>> net=train(net,pn,tn)

>> net

net =

Neural Network object:

architecture:

numInputs: 1

numLayers: 3

biasConnect: [1; 1; 1]

inputConnect: [1; 0; 0]

layerConnect: [0 0 0; 1 0 0; 0 1 0]

outputConnect: [0 0 1]

numOutputs: 1 (read-only)

numInputDelays: 0 (read-only)

numLayerDelays: 0 (read-only)

subobject structures:

inputs: {1x1 cell} of inputs

layers: {3x1 cell} of layers

outputs: {1x3 cell} containing 1 output

biases: {3x1 cell} containing 3 biases

inputWeights: {3x1 cell} containing 1 input weight

layerWeights: {3x3 cell} containing 2 layer weights

functions:

adaptFcn: 'trains'

divideFcn: (none)

gradientFcn: 'calcgrad'

initFcn: 'initlay'

performFcn: 'mse'

plotFcns: {'plotperform','plottrainstate','plotregression'}

trainFcn: 'traingdm'

parameters:

adaptParam: .passes

divideParam: (none)

gradientParam: (none)

initParam: (none)

performParam: (none)

trainParam: .show, .showWindow, .showCommandLine, .epochs,

.time, .goal, .max\_fail, .lr,

.mc, .min\_grad

weight and bias values:

IW: {3x1 cell} containing 1 input weight matrix

LW: {3x3 cell} containing 2 layer weight matrices

b: {3x1 cell} containing 3 bias vectors

other:

name: ''

userdata: (user information)

>> an=sim(net,pn)

an =

Columns 1 through 6

-0.8093 -0.6047 -0.2183 -0.4274 -0.2343 -0.4204

Columns 7 through 12

-0.2192 -0.5978 -0.6201 -0.2247 -0.2343 -0.4366

Columns 13 through 18

-0.2264 -0.6487 -1.2082 -0.0369 0.1691 -0.4740

Columns 19 through 24

0.3648 -0.2391 -1.0204 -0.6307 -0.0350 0.1661

Columns 25 through 30

-0.6492 2.3389 2.5490 -0.6514 -0.4266 -0.4214

Columns 31 through 36

0.5619 2.5371 2.3409 -0.5759 -0.2290 -0.0269

Columns 37 through 42

2.5259 2.7351 -0.4334 -0.2298 0.3637 0.1659

Columns 43 through 48

-0.2290 -0.6244 -0.8224 -0.4117 -0.0375 -0.0330

Columns 49 through 50

-0.4300 -1.0206

>> a=poststd(an,meant,stdt)

a =

Columns 1 through 5

9.0646 10.1001 12.0552 10.9972 11.9744

Columns 6 through 10

11.0325 12.0509 10.1349 10.0223 12.0229

Columns 11 through 15

11.9744 10.9507 12.0141 9.8774 7.0461

Columns 16 through 20

12.9735 14.0157 10.7616 15.0061 11.9503

Columns 21 through 25

7.9967 9.9683 12.9829 14.0004 9.8750

Columns 26 through 30

24.9958 26.0587 9.8635 11.0011 11.0276

Columns 31 through 35

16.0036 25.9986 25.0059 10.2455 12.0010

Columns 36 through 40

13.0240 25.9417 27.0005 10.9670 11.9973

Columns 41 through 45

15.0002 13.9993 12.0014 10.0004 8.9983

Columns 46 through 50

11.0766 12.9705 12.9932 10.9839 7.9956

>> p1=[12;12;10;13];

>> pn1=trastd(p1,meanp,stdp)

pn1 =

-0.1292

-0.1950

-0.6029

-0.0630

>> an=sim(net,pn1)

an =

0.7021

>> a=poststd(an,meant,stdt)

a =

16.7126

>> p2=[13;15;9;13];

>> pn2=trastd(p2,meanp,stdp)

pn2 =

0.0792

0.4020

-0.7868

-0.0630

>> an=sim(net,pn2)

an =

0.8348

>> a=poststd(an,meant,stdt)

a =

17.3842

>> p3=[12;14;7;12];

>> pn3=trastd(p3,meanp,stdp)

pn3 =

-0.1292

0.2030

-1.1544

-0.2485

>> an=sim(net,pn3)

an =

0.3464

>> a=poststd(an,meant,stdt)

a =

14.9130

>> p4=[26;37;25;24];

>> pn4=trastd(p4,meanp,stdp)

pn4 =

2.7890

4.7807

2.1544

1.9766

>> an=sim(net,pn4)

an =

3.2794

>> a=poststd(an,meant,stdt)

a =

29.7547

>> p5=[27;37;29;25];

>> pn5=trastd(p5,meanp,stdp)

pn5 =

2.9974

4.7807

2.8897

2.1620

>> an=sim(net,pn5)

an =

3.0012

>> a=poststd(an,meant,stdt)

a =

28.3471

>> p6=[11;12;11;11];

>> pn6=trastd(p6,meanp,stdp)

pn6 =

-0.3377

-0.1950

-0.4191

-0.4339

>> an=sim(net,pn6)

an =

-0.4719

>> a=poststd(an,meant,stdt)

a =

10.7719

>> p7=[12;13;12;8];

>> pn7=trastd(p7,meanp,stdp)

pn7 =

-0.1292

0.0040

-0.2353

-0.9901

>> an=sim(net,pn7)

an =

-0.1879

>> a=poststd(an,meant,stdt)

a =

12.2090

>> p8=[12;10;13;16];

>> pn8=trastd(p8,meanp,stdp)

pn8 =

-0.1292

-0.5931

-0.0515

0.4932

>> an=sim(net,pn8)

an =

-0.4347

>> a=poststd(an,meant,stdt)

a =

10.9603

>> p9=[15;9;13;13];

>> pn9=trastd(p9,meanp,stdp)

pn9 =

0.4961

-0.7921

-0.0515

-0.0630

>> an=sim(net,pn9)

an =

-0.1938

>> a=poststd(an,meant,stdt)

a =

12.1792

>> p10=[14;7;12;15];

>> pn10=trastd(p10,meanp,stdp)

pn10 =

0.2877

-1.1902

-0.2353

0.3078

>> an=sim(net,pn10)

an =

-1.6920

>> a=poststd(an,meant,stdt)

a =

4.5982

>> p11=[37;25;24;28];

>> pn11=trastd(p11,meanp,stdp)

pn11 =

5.0818

2.3923

1.9706

2.7183

>> an=sim(net,pn11)

an =

2.4503

>> a=poststd(an,meant,stdt)

a =

25.5590

>> p12=[37;29;25;30];

>> pn12=trastd(p12,meanp,stdp)

pn12 =

5.0818

3.1884

2.1544

3.0891

>> an=sim(net,pn12)

an =

2.6656

>> a=poststd(an,meant,stdt)

a =

26.6486

>> p13=[12;11;11;12];

>> pn13=trastd(p13,meanp,stdp)

pn13 =

-0.1292

-0.3941

-0.4191

-0.2485

>> pn13=trastd(p13,meanp,stdp)

pn13 =

-0.1292

-0.3941

-0.4191

-0.2485

>> an=sim(net,pn13)

an =

-0.3581

>> a=poststd(an,meant,stdt)

a =

11.3480

>> p14=[13;12;8;13];

>> pn14=trastd(p14,meanp,stdp)

pn14 =

0.0792

-0.1950

-0.9706

-0.0630

>> an=sim(net,pn14)

an =

0.7433

>> a=poststd(an,meant,stdt)

a =

16.9212

>>