## 神经网络算法的Python和R实现

例：表4.21是某市2003年～2007年间各年某地交通事故情况，试构造神经网络模型进行回归预测。

表4.21 某市2003年～2007年间某地交通事故情况

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 年份 | 事故起数 | 死亡人数 | 受伤人数 | 直接财产损失（万元） |
| 2003 | 80 | 3 | 50 | 11 |
| 2004 | 90 | 8 | 70 | 12.5 |
| 2005 | 180 | 20 | 120 | 20 |
| 2006 | 140 | 16 | 90 | 18 |
| 2007 | 120 | 5 | 85 | 15.5 |

说明：利用前四行数据进行神经网络的回归拟合，第五行数据进行预测

Python代码：

import numpy as np

import tensorflow as tf

x = [[80,3,50],[90,8,70],[180,20,120],[140,16,90]]

y = [[11],[12.5],[20],[18]]

x\_pred = [[120,5,85]]

tf\_x = tf.placeholder(tf.float32, [None,3]) # input x

tf\_y = tf.placeholder(tf.float32, [None,1]) # input y

# neural network layers

l1 = tf.layers.dense(tf\_x, 10, tf.nn.relu) # hidden layer

output = tf.layers.dense(l1, 1) # output layer

loss = tf.losses.mean\_squared\_error(tf\_y, output)# compute cost

optimizer = tf.train.GradientDescentOptimizer(learning\_rate=0.1)

train\_op = optimizer.minimize(loss)

sess = tf.Session() # control training and others

sess.run(tf.global\_variables\_initializer()) # initialize var in graph

for step in range(150):

# train and net output

\_, l, pred = sess.run([train\_op, loss, output], {tf\_x: x, tf\_y: y})

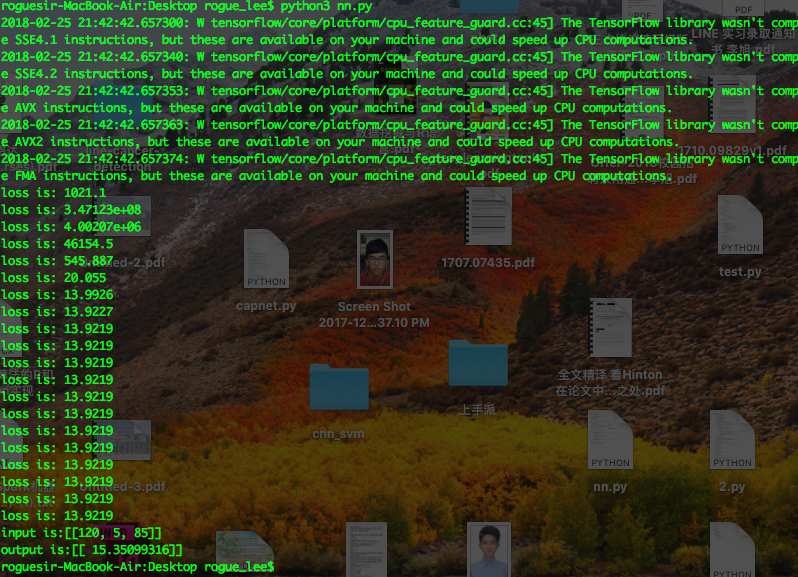
if step % 10 == 0:

print('loss is: ' + str(l))

output\_pred = sess.run(output,{tf\_x:x\_pred})

print('input is:' + str(x\_pred[0][:]))

print('output is:' + str(output\_pred[0][0]))



R代码：

> library(grid) #加载相应的程序包

> library(MASS)

> library(neuralnet)

> m> t <- c(11.0,12.5,20.0,18.0,15.5)

> trainingdata <- cbind(p,t) #合并数据集

> trainingdata

t

[1,] 80 90 180 11.0

[2,] 140 120 3 12.5

[3,] 8 20 16 20.0

[4,] 5 50 70 18.0

[5,] 120 90 85 15.5

> colnames(trainingdata) <- c("Input1","Input2","Input3","Output")

> net <- neuralnet(Output~Input1+Input2+Input3,trainingdata, hidden=20, threshold=0.005, learningrate = 0.1, algorithm = "rprop+", err.fct = "sse", act.fct = "logistic") #建立神经网络处理回归问题，三个输入，一个输出，隐含层20个，阈值为0.005，学习率为0.1，选用rprop+方法进行优化，损失函数为SSE，激活函数为logistic

> print(net)

$call

neuralnet(formula = Output ~ Input1 + Input2 + Input3, data = trainingdata, hidden = 20, threshold = 0.005, learningrate = 0.1, algorithm = "rprop+", err.fct = "sse", act.fct = "logistic")

$response

Output

1 11.0

2 12.5

3 20.0

4 18.0

$covariate

[,1] [,2] [,3]

[1,] 80 90 180

[2,] 140 3 8

[3,] 20 16 50

[4,] 70 120 90

$model.list

$model.list$response

[1] "Output"

$model.list$variables

[1] "Input1" "Input2" "Input3"

$err.fct

function (x, y)

{

1/2 \* (y - x)^2

}

<environment: 0x102f9e1b8>

attr(,"type")

[1] "sse"

$act.fct

function (x)

{

1/(1 + exp(-x))

}

<environment: 0x102f9e1b8>

attr(,"type")

[1] "logistic"

$linear.output

[1] TRUE

$data

Input1 Input2 Input3 Output

1 80 90 180 11.0

2 140 3 8 12.5

3 20 16 50 20.0

4 70 120 90 18.0

$net.result

$net.result[[1]]

[,1]

1 11.00232052

2 12.50002038

3 20.00008036

4 17.99886830

$weights

$weights[[1]]

$weights[[1]][[1]]

[,1] [,2] [,3] [,4] [,5]

[1,] 1.6787200700 1.69219307281 1.0207270442 1.3404950968 0.8895626971

[2,] -1.2889558377 0.75317305849 0.4780492659 0.2785465296 0.8543375425

[3,] 1.6326766717 1.65672266013 -1.7212605757 0.9788560557 1.9458685306

[4,] -0.3534786524 -0.06240399969 3.2040673527 -0.7049333799 -0.1257822759

[,6] [,7] [,8] [,9] [,10]

[1,] 1.9295087655 0.09918449006 1.3857686534 -0.1340420811 -1.24075943724

[2,] -0.1529686319 0.24702967483 -0.8439012631 -0.8257717835 0.66284015949

[3,] 1.6817432112 -0.20176537396 -0.3186500297 0.6270405634 -0.06380447216

[4,] 1.8877998638 0.90175231680 0.4923219499 1.2246106937 0.68943778185

[,11] [,12] [,13] [,14] [,15]

[1,] -0.9905653795 -0.7722678457 1.7265675060 2.8239214212 -0.7147773507

[2,] -0.8398289331 2.3161599745 3.0082378761 -0.7107452348 -1.4441980251

[3,] -2.4259416454 0.9091649618 0.6264874563 -0.4547520583 0.4495687387

[4,] 1.4824004756 -0.3381077066 -0.5648201318 -2.1830193719 1.4097913516

[,16] [,17] [,18] [,19] [,20]

[1,] -0.3153566981 0.6838830999 -0.2016530630 -0.1201685352 -0.6878858961

[2,] -0.1819614467 1.0602572665 0.8940469699 0.4209818221 0.9009061084

[3,] -2.7290682313 0.6148585985 1.6912736716 -0.3536390935 -0.2902131402

[4,] -1.6484520472 0.7719125960 1.3395100829 0.7065197367 0.5163222564

$weights[[1]][[2]]

[,1]

[1,] -0.16328508456

[2,] 4.66191458852

[3,] 2.41330470701

[4,] 0.51500078741

[5,] 2.34193635959

[6,] 1.14521690082

[7,] 1.67001461327

[8,] 0.60461037420

[9,] 3.55870895271

[10,] -0.55886509858

[11,] 1.40782574322

[12,] 5.50877415804

[13,] 1.49236211177

[14,] -0.05570169271

[15,] -0.08484456966

[16,] 0.82733169173

[17,] 0.16072203437

[18,] 0.17904345650

[19,] 0.45250538533

[20,] 0.43355421267

[21,] 0.63209924210

$startweights

$startweights[[1]]

$startweights[[1]][[1]]

[,1] [,2] [,3] [,4] [,5]

[1,] 0.2256350845 0.5921930728 1.0207270442 -0.75342839877 0.08956269713

[2,] -2.1471452861 -0.3468269415 0.4780492659 0.06567268305 0.05433754248

[3,] 0.7888413294 0.5567226601 -1.7212605757 0.78796265543 1.14586853061

[4,] -1.2190922638 -1.1624039997 3.2040673527 -0.91936228078 -0.92578227593

[,6] [,7] [,8] [,9] [,10]

[1,] 1.0108342341 -0.16921550994 1.2294591096 -0.5925904002 -1.24075943724

[2,] -1.0716431632 -0.02137032517 -0.8122402155 -1.2843201026 0.66284015949

[3,] 0.7630686799 -0.47016537396 -0.2659598160 0.1684922443 -0.06380447216

[4,] 0.9691253324 0.63335231680 0.5239829975 0.7660623745 0.68943778185

[,11] [,12] [,13] [,14] [,15]

[1,] -1.848625725 -1.4722678457 0.9581675060 3.7425959525 -1.6334518820

[2,] -1.303812552 1.6161599745 2.2398378761 0.2079292965 -2.3628725564

[3,] -2.880551601 0.2091649618 -0.1419125437 0.4639224730 -0.4691057926

[4,] 1.018234607 -1.0381077066 -1.3332201318 -1.2643448406 0.4911168203

[,16] [,17] [,18] [,19] [,20]

[1,] 0.3560289139 0.6838830999 -0.2016530630 -0.5885685352 -0.7378858961

[2,] 0.4894241653 1.0602572665 0.8940469699 -0.0474181779 0.8509061084

[3,] -2.0576826194 0.6148585985 1.6912736716 -0.8220390935 -0.3402131402

[4,] -0.9770664352 0.7719125960 1.3395100829 0.2381197367 0.4663222564

$startweights[[1]][[2]]

[,1]

[1,] -0.85650133713

[2,] -0.13660173422

[3,] 1.72008845444

[4,] -0.17821546517

[5,] -0.07765372835

[6,] 0.45200064825

[7,] 1.24936245312

[8,] -0.08860587838

[9,] -0.70367796231

[10,] -0.20609187920

[11,] 0.71460949064

[12,] 0.88789309191

[13,] 0.79914585919

[14,] -0.66716450013

[15,] -1.00351910099

[16,] 1.18010491111

[17,] -0.75795249695

[18,] -0.51417279608

[19,] -0.24071086725

[20,] -0.25966203990

[21,] -0.06111701048

$generalized.weights

$generalized.weights[[1]]

[,1] [,2]

1 0.000055883032513374445945410207148 0.000021095240405134215745924092245

2 0.000399007641796709903482531789365 -0.004386705851859396682046465798521

3 0.000136204016641501963981106260171 0.000051409232087452285829467374390

4 0.000000000000000000000000376031181 0.000000000000000000000000141986216

[,3]

1 -0.0000325980174477019765851004695634

2 -0.0049241897660020706242733901092379

3 -0.0000794475209487746376196826147975

4 -0.0000000000000000000000002193721142

$result.matrix

1

error 0.00000333622074

reached.threshold 0.00329037073866

steps 98.00000000000000

Intercept.to.1layhid1 1.67872006995493

Input1.to.1layhid1 -1.28895583768178

Input2.to.1layhid1 1.63267667174304

Input3.to.1layhid1 -0.35347865240345

Intercept.to.1layhid2 1.69219307281354

Input1.to.1layhid2 0.75317305849236

Input2.to.1layhid2 1.65672266012814

Input3.to.1layhid2 -0.06240399968626

Intercept.to.1layhid3 1.02072704420385

Input1.to.1layhid3 0.47804926591620

Input2.to.1layhid3 -1.72126057574062

Input3.to.1layhid3 3.20406735266928

Intercept.to.1layhid4 1.34049509682303

Input1.to.1layhid4 0.27854652961478

Input2.to.1layhid4 0.97885605572280

Input3.to.1layhid4 -0.70493337992833

Intercept.to.1layhid5 0.88956269712896

Input1.to.1layhid5 0.85433754247725

Input2.to.1layhid5 1.94586853060589

Input3.to.1layhid5 -0.12578227593212

Intercept.to.1layhid6 1.92950876545225

Input1.to.1layhid6 -0.15296863192167

Input2.to.1layhid6 1.68174321118293

Input3.to.1layhid6 1.88779986377255

Intercept.to.1layhid7 0.09918449005615

Input1.to.1layhid7 0.24702967483214

Input2.to.1layhid7 -0.20176537396359

Input3.to.1layhid7 0.90175231679586

Intercept.to.1layhid8 1.38576865338002

Input1.to.1layhid8 -0.84390126307261

Input2.to.1layhid8 -0.31865002970199

Input3.to.1layhid8 0.49232194989809

Intercept.to.1layhid9 -0.13404208108405

Input1.to.1layhid9 -0.82577178346967

Input2.to.1layhid9 0.62704056344510

Input3.to.1layhid9 1.22461069365879

Intercept.to.1layhid10 -1.24075943724086

Input1.to.1layhid10 0.66284015949123

Input2.to.1layhid10 -0.06380447216320

Input3.to.1layhid10 0.68943778184774

Intercept.to.1layhid11 -0.99056537951018

Input1.to.1layhid11 -0.83982893310833

Input2.to.1layhid11 -2.42594164535841

Input3.to.1layhid11 1.48240047560707

Intercept.to.1layhid12 -0.77226784571603

Input1.to.1layhid12 2.31615997450187

Input2.to.1layhid12 0.90916496182599

Input3.to.1layhid12 -0.33810770661700

Intercept.to.1layhid13 1.72656750603806

Input1.to.1layhid13 3.00823787608971

Input2.to.1layhid13 0.62648745626868

Input3.to.1layhid13 -0.56482013184052

Intercept.to.1layhid14 2.82392142117465

Input1.to.1layhid14 -0.71074523484274

Input2.to.1layhid14 -0.45475205833266

Input3.to.1layhid14 -2.18301937190443

Intercept.to.1layhid15 -0.71477735071010

Input1.to.1layhid15 -1.44419802506314

Input2.to.1layhid15 0.44956873868148

Input3.to.1layhid15 1.40979135161913

Intercept.to.1layhid16 -0.31535669806291

Input1.to.1layhid16 -0.18196144668368

Input2.to.1layhid16 -2.72906823134883

Input3.to.1layhid16 -1.64845204716267

Intercept.to.1layhid17 0.68388309990236

Input1.to.1layhid17 1.06025726647877

Input2.to.1layhid17 0.61485859852038

Input3.to.1layhid17 0.77191259600945

Intercept.to.1layhid18 -0.20165306295925

Input1.to.1layhid18 0.89404696993542

Input2.to.1layhid18 1.69127367159548

Input3.to.1layhid18 1.33951008290973

Intercept.to.1layhid19 -0.12016853521657

Input1.to.1layhid19 0.42098182210182

Input2.to.1layhid19 -0.35363909345263

Input3.to.1layhid19 0.70651973665609

Intercept.to.1layhid20 -0.68788589614646

Input1.to.1layhid20 0.90090610844781

Input2.to.1layhid20 -0.29021314022957

Input3.to.1layhid20 0.51632225641782

Intercept.to.Output -0.16328508455836

1layhid.1.to.Output 4.66191458851900

1layhid.2.to.Output 2.41330470701275

1layhid.3.to.Output 0.51500078741141

1layhid.4.to.Output 2.34193635958627

1layhid.5.to.Output 1.14521690082260

1layhid.6.to.Output 1.67001461327046

1layhid.7.to.Output 0.60461037419848

1layhid.8.to.Output 3.55870895271396

1layhid.9.to.Output -0.55886509858291

1layhid.10.to.Output 1.40782574321972

1layhid.11.to.Output 5.50877415804306

1layhid.12.to.Output 1.49236211177114

1layhid.13.to.Output -0.05570169270910

1layhid.14.to.Output -0.08484456966379

1layhid.15.to.Output 0.82733169172856

1layhid.16.to.Output 0.16072203437372

1layhid.17.to.Output 0.17904345649595

1layhid.18.to.Output 0.45250538532949

1layhid.19.to.Output 0.43355421267165

1layhid.20.to.Output 0.63209924209802

attr(,"class")

[1] "nn"

> plot(net) # 画网络图

> testdata <- matrix(c(120.0,5.0,85.0),1,3,byrow = T) # 建立测试集

> net.results <- compute(net, testdata) # 利用训练好的网络进行预测

> ls(net.results)

[1] "net.result" "neurons"

> print(net.results$net.result) # 输出预测值

[,1]

[1,] 15.67661666



注：

（1）蓝色字为实现代码，黑色字为代码执行结果。

（2）系统环境：MacOS，Python3.6，TensorFlow1.1.0

（3）Python代码：如果没有安装TensorFlow库，需要在terminal中执行pip3 install tensorflow命令进行安装

（4）R代码：需要提前安装grid、MASS、neuralnet算法包