丁香园

丁香公开课
CLASS.DXY.CN

临床预测模型构建&机器学习(R语言进阶)

第15章 神经网络与深度学习在医学中应用

周支瑞

CONTENT

01 | 神经网络原理

02 | 深度学习原理

03 | 深度神经网络举例

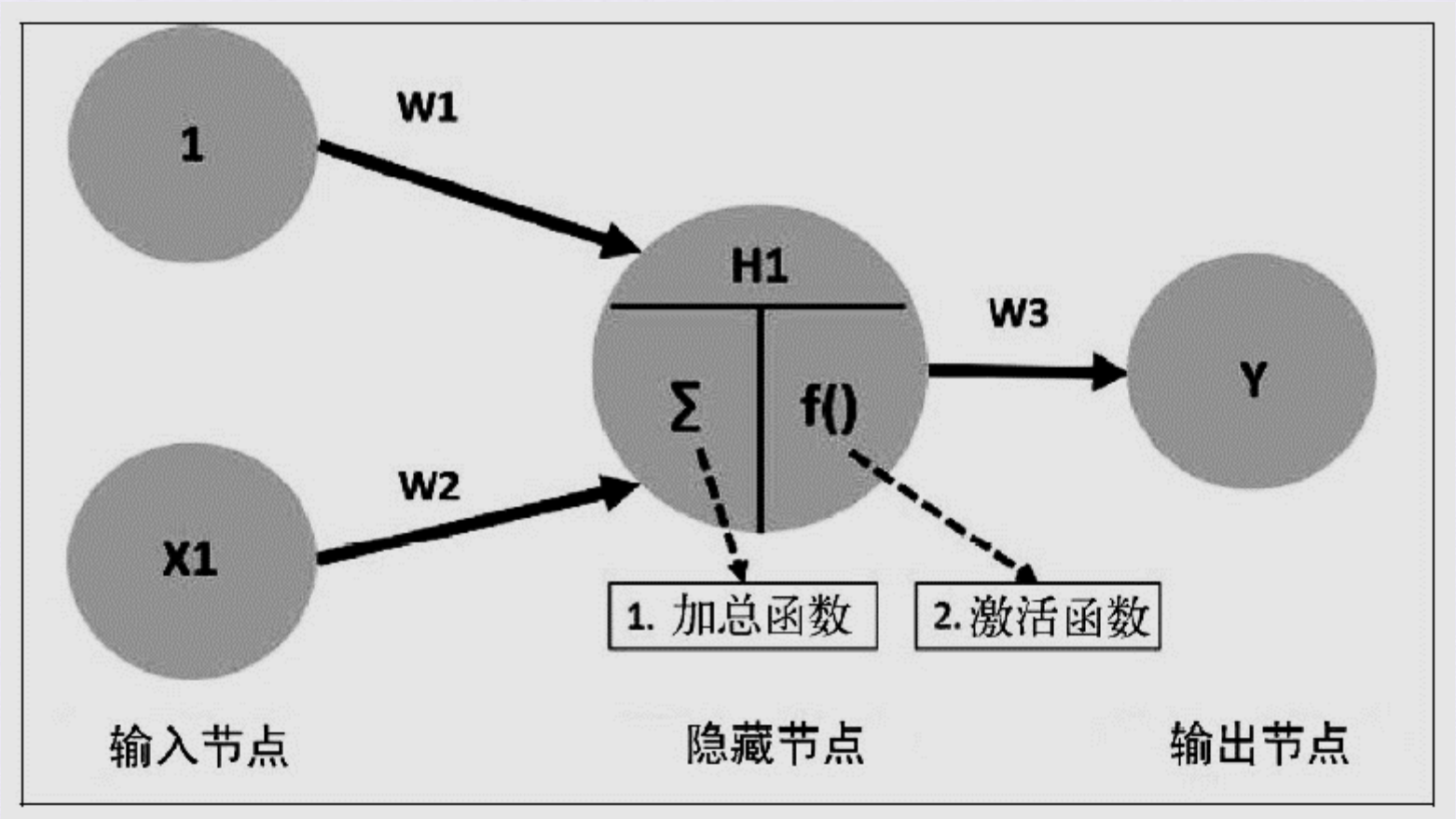
04 | 神经网络案例分析

05 | 深度学习案例分析

神经网络 概念

- 在下图这个简单的网络中，输入（又称协变量）由两个节点（又称神经元）组成。标有1的神经元代表一个常数，更确切地说，是截距。X1代表一个定量变量，W代表权重，会和输入节点的值相乘。输入节点的值在与W相乘后传递到隐藏节点。你可以有多个隐藏节点，但是工作原理和一个隐藏节点没有什么不同。在隐藏节点H1中，所有加权后的输入值被加总。因为截距为1，所以这个输入节点的值就是权重W1。加总后的值通过**激活函数**进行转换，将输入信号转换为输出信号。在这个简单例子中，H1是唯一的隐藏节点，它的值被乘以W3，成为响应变量Y的估计值。这就是算法的前馈部分。

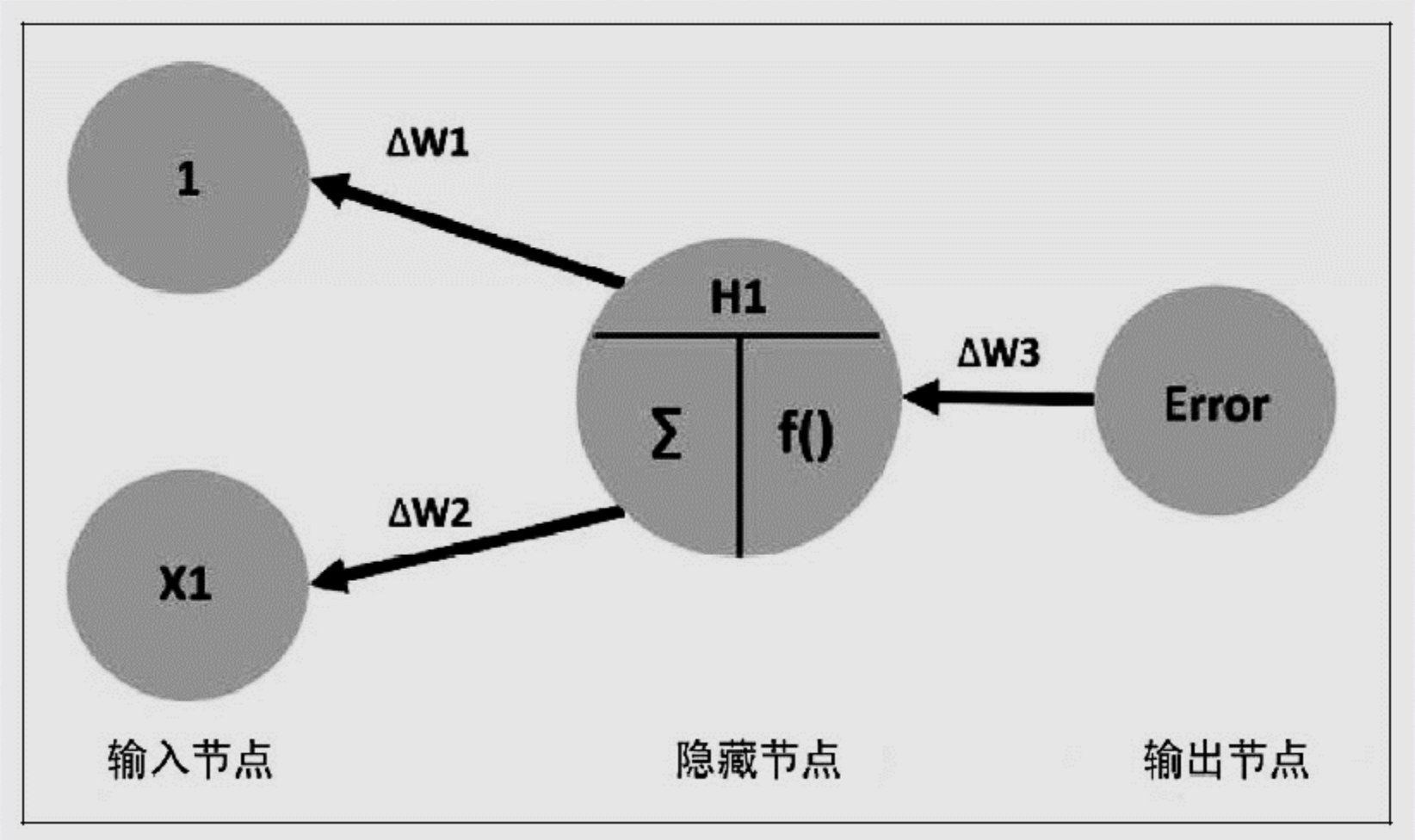
神经网络 前馈部分



神经网络 完整循环

- 要完成这个循环，或者用神经网络中的说法，完成一次完整的“训练”，还要进行反向传播过程，基于学习到的知识来训练模型。为了初始化反向传播过程，需要基于损失函数确定误差，损失函数可以是误差平方总和，也可以是交叉熵，或者其他形式。因为权重W1和W2最初被设定为[-1, 1]之间的随机数，所以初始的误差可能会很大。反向传播时，要改变权重值以使损失函数中的误差最小。下页的图描述了算法的反向传播部分。

神经网络 反向传播



神经网络

- 这样就完成了一次完整的训练。这个过程不断继续，使用梯度下降方法可减小误差，直到算法收敛于误差最小值或者达到预先设定的训练次数。如果假设本例中的激活函数是简单线性的，那么结果为Y = W3(W1(1) + W2(X1))。
- 如果你增加了大量的输入神经元，或者在隐藏节点中增加多个神经元，甚至增加多个隐藏节点，神经网络会变得非常复杂。请一定注意，一个神经元的输出会连接到所有后继的神经元，并将权重分配给所有连接，这样会大大增加模型的复杂性。增加隐藏节点和提高隐藏节点中神经元的数量不会像我们希望的那样改善模型的性能。于是，深度学习得以发展，它可以部分放松所有神经元都要连接的要求。

激活函数

- 有很多种激活函数可供使用，其中包括一个简单线性函数，还有用于分类问题的sigmoid函数，它是Logistic函数的一种特殊形式。当输出变量是基于某种阈值的二值变量（0或1）时，可以使用阈值函数。其他常用的激活函数还有Rectifier、Maxout以及双曲正切函数（tanh）。

代码如下


```
sigmoid <- function(x) {
  1 / ( 1 + exp(-x) )
  }
x <- seq(-5, 5, .1)
plot(sigmoid(x))

library(ggplot2)
s <- sigmoid(x)
t <- tanh(x)
z <- data.frame(cbind(x, s, t))
ggplot(z, aes(x)) +
 geom_line(aes(y = s, color = "sigmoid")) +
 geom_line(aes(y = t, color = "tanh")) +
 labs(x = "Input",y = "Output")
```

CONTENT

01 | 神经网络原理

02 | 深度学习原理

03 | 深度神经网络举例

04 | 神经网络案例分析

05 | 深度学习案例分析

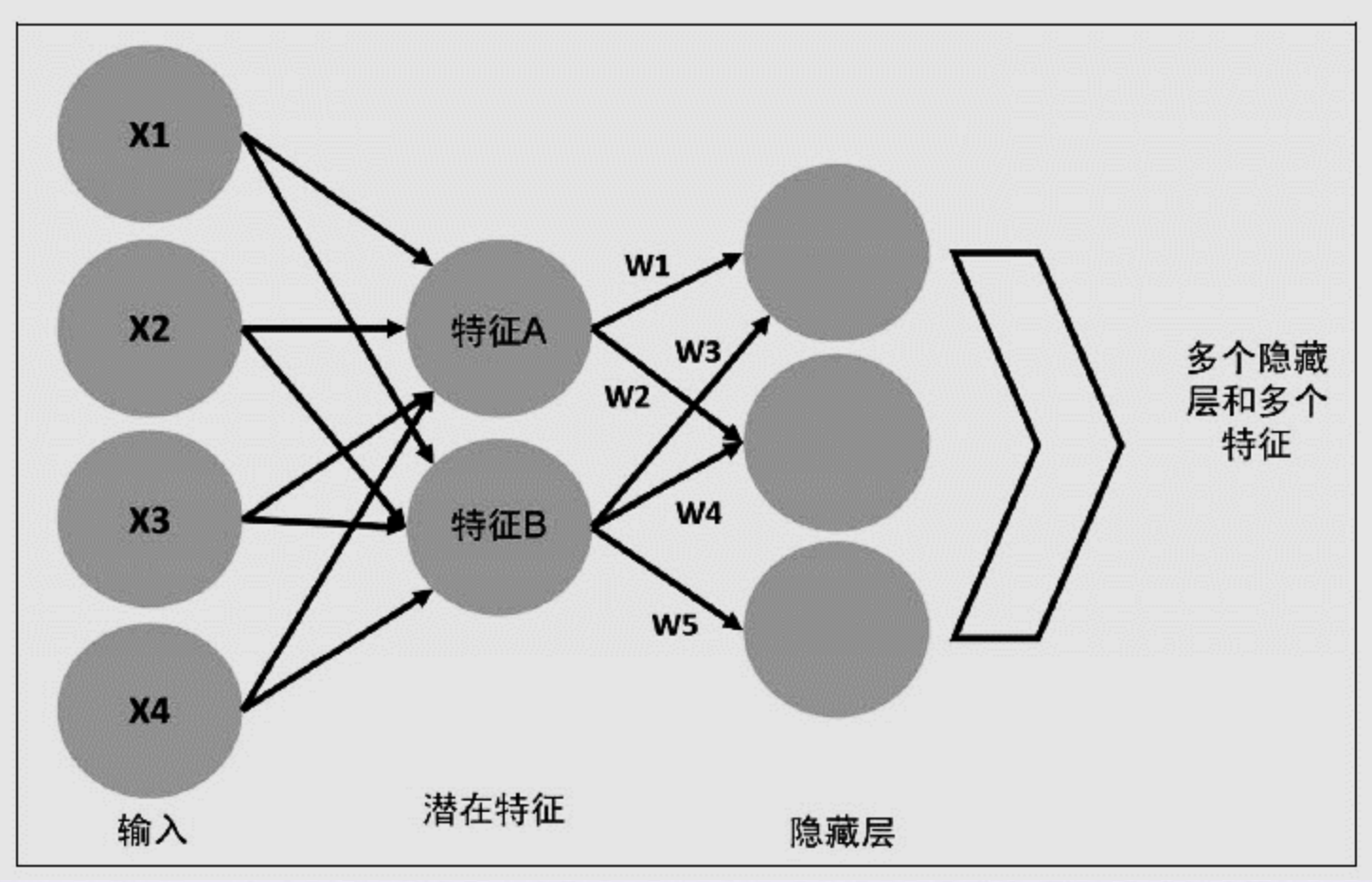
深度学习 概念

- 深度学习基础是神经网络，它的特点就是使用机器学习技术（一般是无监督学习）在输入变量的基础之上构建新的特征。可以认为它们就是在没有响应变量的情况下寻找数据中的结构。元素周期表就是一个经典例子，它在没有指定响应变量时找出数据结构。你展开元素周期表就会看到，它是按照原子结构排列，金属在一侧，非金属在另一侧，基于潜在分类（结构）建立。这种对潜在结构（层次）的识别就是深度学习与那些神经网络的区别所在。相比于那些只有原始输入的问题，深度学习更适用于那种可以通过算法对输出结果进行更好表示的问题。换句话说，模型在仅有原始像素作为唯一输入的情况下，能够学会对照片进行分类吗？当你有少量标记过的响应变量和海量未标记的输入数据时，深度学习非常有用。你可以通过无监督学习方法训练深度学习模型，然后通过监督式方法将其应用在标记过的数据上，这样向前/向后不断迭代。

深度学习 概念

- 要用数学解释这种潜在结构的识别是非常有难度的，但我们在第5章介绍过的正则化概念正是一个例子。在深度学习中，你可以使用正则化方法对权重进行惩罚，比如L1（惩罚非0权重）、L2（惩罚过大权重）和丢弃（随机忽略某种输入，将其权重归零）。标准神经网络中不使用这些正则化技术。
- 另外一种识别潜在结构的方式是降低数据的维度，比如自动编码器。在这种神经网络中，输入被转换为一组降低了维度的权重。请注意，下页图中的特征A和一个隐藏节点之间没有连接。

深度学习 示意图



深度学习

- 这种方法可以递归使用，学习可以在多个隐藏层之间进行。在这个例子中能发现的就是，网络在不断使用原有特征制造新特征，以至于它们互相堆叠在一起。深度学习会先在两个层之间按前后顺序学习权重，然后使用反向传播方法对这些权重进行微调。其他的特征选择方法包括受限波尔兹曼机和稀疏编码模型。

深度学习

深度学习在很多分类问题上表现得非常好，和神经网络一样，它也受到一些问题困扰，最主要的就是黑盒问题。你可以试着向不明情况的人解释一下神经网络中都发生了什么。尽管如此，对于那种“不管黑猫白猫，捉住老鼠就是好猫”的问题，深度学习是非常适合的。归根结底，我们真的想知道自动驾驶汽车为什么可以避开行人吗？或者说，我们更关注的难道不是自动驾驶汽车是否撞到了行人吗？Python在深度学习的使用 和软件开发方面领先R一步。我们在后面的实战练习中会看到，这个距离正在逐渐缩小，但还没有完全消除。

深度学习

- 不管怎么说，深度学习都是一种激动人心的技术。应该注意的是，要想发挥这项技术的全部威力，你不但需要强大的计算能力，还需要花费大量时间，通过微调超参数去训练最优模型。以下是一些注意事项：
  （1）激活函数
  （2）隐藏层的规模
  （3）降低维度的方法（受限波尔兹曼机还是自动编码器）
  （4）完整训练的次数
  （5）梯度下降学习率
  （6）损失函数
  （7）正则化

CONTENT

01 | 神经网络原理

02 | 深度学习原理

03 | 深度神经网络 举例

04 | 神经网络案例分析

05 | 深度学习案例分析

卷积神经网络

- 前提假设是输入为图像，并使用一小部分数据或数据切片来创建特征，并将特征组合创建特征映射图。这些小数据切片可以作为网络在训练过程中学习的过滤器，或者更确切地说，是卷积核。CNN的激活函数是线性整流函数，它的简单形式是f(x)=max(0,x)，其中x是神经元的输入。CNN在图像分类、目标检测甚至句子分类方面的效果非常好。

循环神经网络

- 创建RNN的目的是利用序贯信息。在传统神经网络中，输入与输出之间是互相独立的。而在RNN中，输出则依赖于上一层的计算结果，允许信息在层与层之间保持。所以，对于一个神经元的输出（y），并不仅是根据它的输入计算出来的，还要包括所有上层的输出（t1,t2,tn…）。RNN对于笔迹检测和语音检测特别有效。

长短期记忆网络

- LSTM是RNN的一种特殊形式。RNN有一个问题，它在具有长期信号的数据上表现不佳。于是，数据科学家们创建了LSTM来捕获数据中的复杂模式。在训练过程中，RNN组合信息时，对来自以前步骤的信息以同样的方式进行处理，没有考虑不同步骤中的信息在价值上多少会有些不同。LSTM通过确定在训练过程中的每个步骤要记住哪些信息，以此克服RNN的这种局限性。这时的权重矩阵通过数据向量实现了倍增，在LSTM中称为“门”，它的作用就是过滤信息。LSTM中的神经元有两个输入和两个输出，它接受来自上一层神经元的输出和从前一个门传递过来的记忆向量，然后生成输出值和输出记忆向量，作为下一层的输入。LSTM的局限性在于，它需要完整清晰的训练数据以及非常高的计算能力。LSTM在语音识别问题上的效果非常好。

CONTENT

01 | 神经网络原理

02 | 深度学习原理

03 | 深度神经网络举例

04 | 神经网络案例分析

05 | 深度学习案例分析

神经网络 案例1

- MASS包中有shuttle数据集，在这个案例中，我们将开发一个神经网络模型，来回答“航天飞机是否应该使用自动着陆系统”这一问题。默认的决策是让机组人员控制飞船着陆，但在以下两种情况下需要自动着陆系统：一是当机组人员失去控制飞船的能力时；二是当飞船脱离轨道之后重新进入轨道，从而需要对抗重力时。数据是由计算机模拟产生的，不是从实际的飞行过程中得来的。实际上，自动着陆系统通过了一些非常严格的测试，绝大多数情况下，在飞船着陆过程中，由宇航员决定是否使用自动着陆系统。

shuttle数据集

- 数据集包括256个观测和7个变量。请注意，所有变量都是分类变量，响应变量use有两个水平，auto和noauto。协变量如下所示：
  （1）stability：能否稳定定位（stab/xstab）；
  （2）error：误差大小（MM/SS/LX）；
  （3）sign：误差的符号，正或负（pp/nn）；
  （4）wind：风向的符号（head/tail）；
  （5）magn：风力强度（Light/Medium/Strong/Out of Range）
  （6）vis：能见度（yes/no）

数据准备 代码

```
library(caret)
library(MASS)
library(neuralnet)
library(vcd)

data(shuttle)
str(shuttle)
table(shuttle$use)
table1 <- structable(wind + magn ~ use, shuttle)
table1
mosaic(table1, shade = T)
mosaic(use ~ error + vis, shuttle)
table(shuttle$use, shuttle$stability)
prop.table(table(shuttle$use, shuttle$stability))
chisq.test(shuttle$use, shuttle$stability)
```

数据准备 代码续


```
dummies <- dummyVars(use ~. ,shuttle, fullRank = T)
dummies
shuttle.2 <- data.frame(predict(dummies, newdata = shuttle))
names(shuttle.2)
head(shuttle.2)
shuttle.2$use <- ifelse(shuttle$use == "auto", 1, 0)
table(shuttle.2$use)
set.seed(123)
trainIndex <- createDataPartition(shuttle.2$use, p = .7,
                                  list = F)
# head(trainIndex)
shuttleTrain <- shuttle.2[ trainIndex, ]
shuttleTest  <- shuttle.2[-trainIndex, ]
```

模型构建与评价

- 前面提到过，我们要使用neuralnet包构建模型，其中建模函数使用的公式和其他方法一样，例如y ~ x1 + x2 + x3 + x4, data = df。以前，我们使用y~.指定数据集中除响应变量之外的所有变量作为输入，但neuralnet中不允许这种写法。绕过这种限制的方式是，使用as.formula()函数。先建立一个保存变量名的对象，然后用这个对象作为输入，从而将变量名粘贴到公式右侧

模型构建与评价

- 在nerualnet包中，我们要使用的函数的名字就叫作nerualnet()。除了模型公式，还有4个关键参数需要说明。
  （1）hidden：每层中隐藏神经元的数量，最多可以设置3个隐藏层，默认值为1。
  （2）act.fct：激活函数，默认为逻辑斯蒂函数，也可以设置为tanh函数。
  （3）err.fct：计算误差，默认为sse；因为我们处理的是二值结果变量，所以要设置成ce，使用交叉熵。
  （4）linear.output：逻辑参数，控制是否忽略act.fct，默认值为TRUE；对于我们的数据来说，需要设置为FALSE。
- 还可以指定算法，默认算法是弹性反向传播。我们就使用这种算法，隐藏神经元也一样，使用默认值1

模型构建与评价 代码

```
n <- names(shuttleTrain)
form <- as.formula(paste("use ~", paste(n[!n %in% "use"], collapse = " + ")))
form
set.seed(1)
fit <- neuralnet(form, data = shuttleTrain, hidden = c(2, 1), err.fct = "ce",
                 linear.output = F)
fit$result.matrix
head(fit$generalized.weights[[1]])
plot(fit)
par(mfrow=c(1,2))
gwplot(fit, selected.covariate = "vis.yes")
gwplot(fit, selected.covariate = "wind.tail")
```

模型构建与评价 代码续


```
resultsTrain <- compute(fit, shuttleTrain[, 1:10])
predTrain <- resultsTrain$net.result
predTrain <- ifelse(predTrain >= 0.5, 1, 0)
table(predTrain, shuttleTrain$use)

resultsTest <- compute(fit, shuttleTest[,1:10])
predTest <- resultsTest$net.result
predTest <- ifelse(predTest >= 0.5, 1, 0)
table(predTest, shuttleTest$use)

which(predTest == 1 & shuttleTest$use == 0)
shuttleTest[62,]
```

神经网络 案例2

- 下面我们用神经网络的方法来解决第3章中列举的有关乳腺癌的问题。这个数据集在可以在R的MASS包中找到该数据框，名为biopsy。在另外一个机器学习的包mlbench中，该数据集被命名为BreastCancer.

BreastCancer.数据集

- 该数据集包含699观测，11个变量，一个字符串变量，9个等级或分类变量，一个结局变量。如下：
  [,1] Id Sample code number
  [,2] Cl.thickness Clump Thickness
  [,3] Cell.size Uniformity of Cell Size
  [,4] Cell.shape Uniformity of Cell Shape
  [,5] Marg.adhesion Marginal Adhesion
  [,6] Epith.c.size Single Epithelial Cell Size
  [,7] Bare.nuclei Bare Nuclei
  [,8] Bl.cromatin Bland Chromatin
  [,9] Normal.nucleoli Normal Nucleoli
  [,10] Mitoses Mitoses [,11] Class Class

数据准备 代码

```
library(mlbench)
library(neuralnet)
data(BreastCancer)
summary(BreastCancer)
mvindex = unique (unlist (lapply (BreastCancer, function (x) which (is.na (x)))))
mvindex
data_cleaned <- na.omit(BreastCancer)
summary(data_cleaned)
boxplot(data_cleaned[,2:10])
hist(as.numeric(data_cleaned$Mitoses))
par(mfrow=c(3, 3))
hist(as.numeric(data_cleaned$Cl.thickness))
hist(as.numeric(data_cleaned$Cell.size))
hist(as.numeric(data_cleaned$Cell.shape))
hist(as.numeric(data_cleaned$Marg.adhesion))
hist(as.numeric(data_cleaned$Epith.c.size))
hist(as.numeric(data_cleaned$Bare.nuclei))
hist(as.numeric(data_cleaned$Bl.cromatin))
hist(as.numeric(data_cleaned$Normal.nucleoli))
hist(as.numeric(data_cleaned$Mitoses))
```

数据准备 代码续

```
str(data_cleaned)
input<-data_cleaned[,2:10]
indx <- sapply(input, is.factor)
input <- as.data.frame(lapply(input, function(x) as.numeric(as.character(x))))
max_data <- apply(input, 2, max)
min_data <- apply(input, 2, min)
input_scaled <- as.data.frame(scale(input,center = min_data, scale = max_data - min_data))
View(input_scaled)
Cancer<-data_cleaned$Class
Cancer<-as.data.frame(Cancer)
Cancer<-with(Cancer, data.frame(model.matrix(~Cancer+0)))
final_data<-as.data.frame(cbind(input_scaled,Cancer))

index = sample(1:nrow(final_data),round(0.70*nrow(final_data)))
train_data <- as.data.frame(final_data[index,])
test_data <- as.data.frame(final_data[-index,])
```

模型构建与评价 代码


```
n = names(final_data[1:9])
f = as.formula(paste("Cancerbenign + Cancermalignant ~", paste(n, collapse = " + ")))

net = neuralnet(f,data=train_data,hidden=5,linear.output=FALSE)
plot(net)

predict_net_test <- compute(net,test_data[,1:9])
predict_result<-round(predict_net_test$net.result, digits = 0)
net.prediction = c("benign", "malignant")[apply(predict_result, 1, which.max)]
predict.table = table(data_cleaned$Class[-index], net.prediction)
predict.table

library(gmodels)
CrossTable(x = data_cleaned$Class[-index], y = net.prediction,
           prop.chisq=FALSE)
```

CONTENT

01 | 神经网络原理

02 | 深度学习原理

03 | 深度神经网络

04 | 神经网络案例分析

05 | 深度学习案例分析

深度学习案例

- 我们要使用的数据是从加州大学欧文分校机器学习库中获取的，我对这份数据进行了一些处理，原始数据及其描述参见https://archive.ics.uci.edu/ml/datasets/Bank+Marketing。我们所做的处理工作包括下载小数据集bank.csv，将数值型变量按比例缩放为均值为0、方差为1，为字符型变量或稀疏数值型变量创建虚拟变量，并删除方差基本为0的变量。这份数据可以在github上找到，地址为https://github.com/datameister66/data/，数据集名称bank_DL.csv。我们在本节着重讨论如何将数据加载到H2O平台，并运行深度学习代码来建立一个分类器，预测客户是否会对一场营销活动做出反应。

H2O平台介绍

- H2O是一个开源的预测分析平台，它有很多预置算法，比如K最近邻、梯度提升机和深度学习。你可以通过Hadoop、AWS、Spark、SQL、noSQL或自己的硬盘将数据上载到平台。H2O的一个巨大优点是，你可以在自己的本地计算机上使用平台上的大多数机器学习算法，这些算法是用R实现的。如果想知道更多细节，请访问站点http://h2o.ai/product/。

深度学习案例 代码

```
# The following two commands remove any previously installed H2O packages for R.
if ("package:h2o" %in% search()) { detach("package:h2o", unload=TRUE) }
if ("h2o" %in% rownames(installed.packages())) { remove.packages("h2o") }

# Next, we download packages that H2O depends on.
if (! ("methods" %in% rownames(installed.packages()))) { install.packages("methods") }
if (! ("statmod" %in% rownames(installed.packages()))) { install.packages("statmod") }
if (! ("stats" %in% rownames(installed.packages()))) { install.packages("stats") }
if (! ("graphics" %in% rownames(installed.packages()))) { install.packages("graphics") }
if (! ("RCurl" %in% rownames(installed.packages()))) { install.packages("RCurl") }
if (! ("jsonlite" %in% rownames(installed.packages()))) { install.packages("jsonlite") }
if (! ("tools" %in% rownames(installed.packages()))) { install.packages("tools") }
if (! ("utils" %in% rownames(installed.packages()))) { install.packages("utils") }

# Now we download, install and initialize the H2O package for R.
install.packages("h2o", type="source", repos=(c("http://h2o-release.s3.amazonaws.com/h2o/rel-
tverberg/5/R")))
library(h2o)

path <- "C:/h2o/bank_DL.csv" # C盘新建h2o文件夹，把bank_DL.csv提前拷贝进去。
localH2O = h2o.init(nthreads = -1)
```

深度学习案例 代码续


```
bank <- h2o.uploadFile(path=path)
class(bank)
str(bank)
head(bank)
summary(bank)
h2o.table(bank$y)

rand <- h2o.runif(bank, seed = 123)

train <- bank[rand <= 0.7, ]
train <- h2o.assign(train, key = "train")
test <- bank[rand  > 0.7, ]
test <- h2o.assign(test, key = "test")

h2o.table(train[, 64])
h2o.table(test[, 64])

hyper_params <- list(
  activation = c("Tanh", "TanhWithDropout"),
  hidden = list(c(20,20),c(40, 40),c(30, 30, 30)),
  input_dropout_ratio = c(0, 0.05),
  rate = c(0.01, 0.25)
)
```

深度学习案例 代码续

```
search_criteria = list(
  strategy = "RandomDiscrete", max_runtime_secs = 420,
  max_models = 100, seed = 123, stopping_rounds = 5,
  stopping_tolerance = 0.01
)

randomSearch <- h2o.grid(
  algorithm = "deeplearning",
  grid_id = "randomSearch",
  training_frame = train,
  validation_frame = test,
  x = 1:63,
  y = 64,
  epochs = 1,
  stopping_metric = "misclassification",
  hyper_params = hyper_params,
  search_criteria = search_criteria
)

grid <- h2o.getGrid("randomSearch", sort_by = "auc", decreasing = T)
grid

best_model <- h2o.getModel(grid@model_ids[[1]])
h2o.confusionMatrix(best_model, valid = T)
```

深度学习案例 代码续


```
dlmodel <- h2o.deeplearning(
  x = 1:63,
  y = 64,
  training_frame = train,
  hidden = c(30, 30, 30),
  epochs = 3,
  nfolds = 5,
  fold_assignment="Stratified",
  balance_classes = T,
  activation = "TanhWithDropout",
  seed = 123,
  adaptive_rate = F,
  input_dropout_ratio = 0.05,
  stopping_metric = "misclassification",
  variable_importances = T
)

dlmodel

perf <- h2o.performance(dlmodel, test)
perf

dlmodel@model$variable_importances
```

请在此处输入小标题

感谢观看

THANKS

丁香园特邀讲师 周支瑞