Chapter 11 – Neural Nets

Instructor: Zach Zhizhong ZHOU,
Shanghai Jiao Tong University
主讲教师：周志中，上海交通大学

Data Mining for Business Intelligence

Shmueli, Patel & Bruce

Basic Idea

- Combine input information in a complex & flexible neural net “model” 将输入信息放在一个复杂而灵活的神经网模型中。

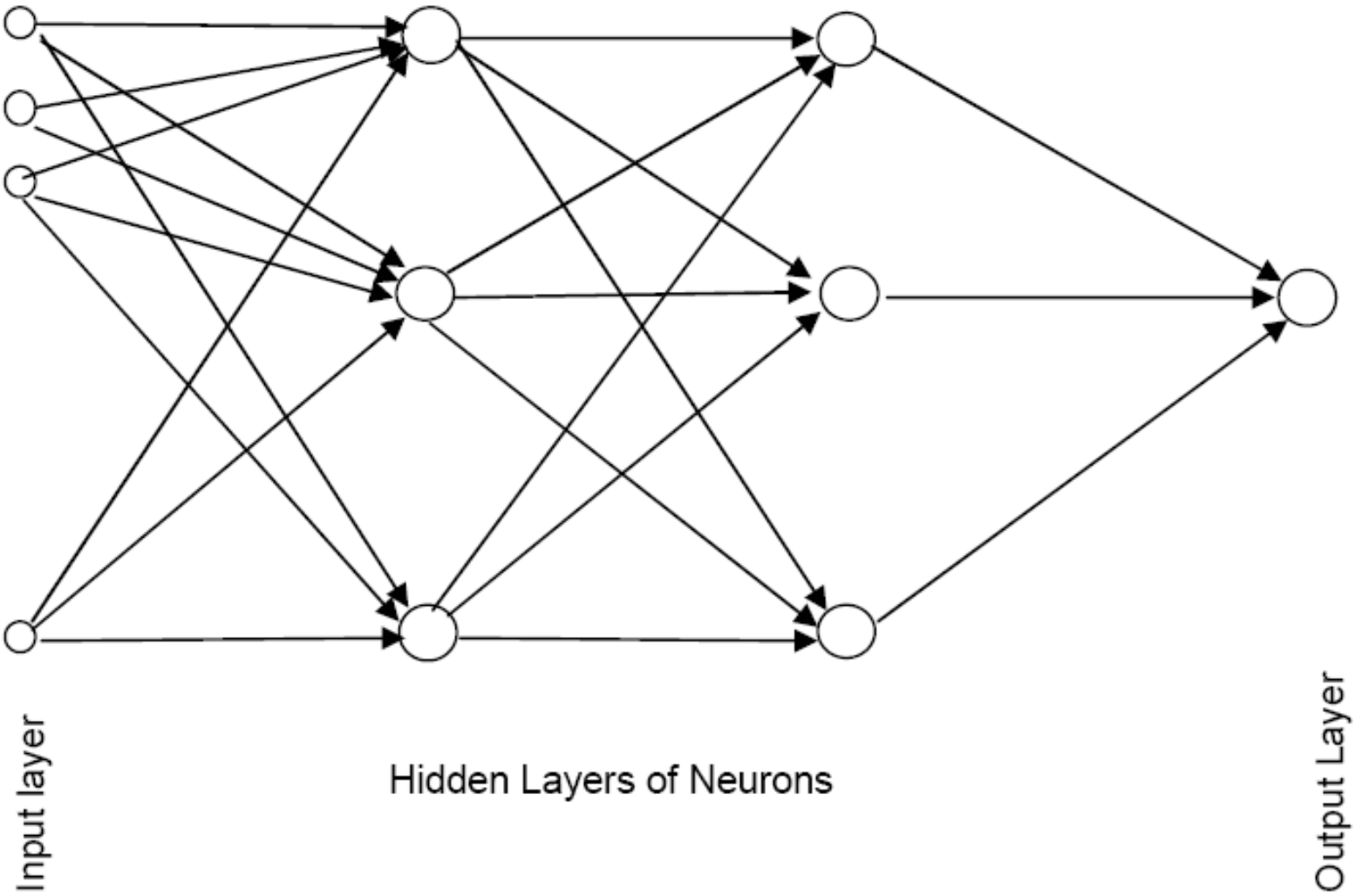
- Model “coefficients” are continually tweaked in an iterative process 模型系数不断反复调整。

- The network’s interim performance in classification and prediction informs successive tweaks 网络模型调整过程中的预测及分类表现作为下次调整的依据。

Network Structure 网络结构

- Multiple layers 多层结构
  - Input layer (raw observations) 输入层 （原始观察值）
  - Hidden layers 隐藏层
  - Output layer 输出层
- Nodes 节点
- Weights (like coefficients, subject to iterative adjustment) 权重（类似参数，需要反复调整）
- Bias values 偏置值

Schematic Diagram



Example – Using fat & salt content to predict consumer acceptance of cheese

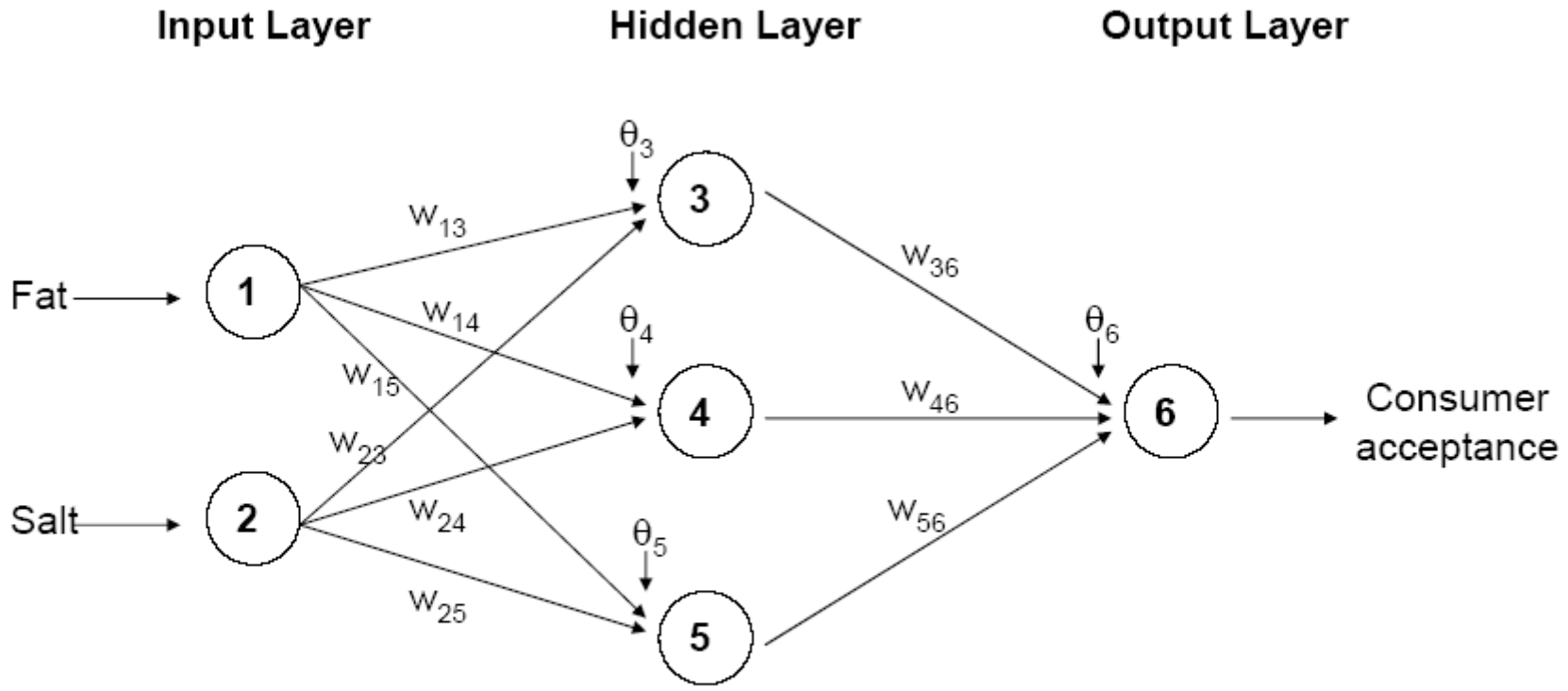


Figure 11.2: Neural network for the tiny example. Circles represent nodes, $w_{i,j}$ on arrows are weights, and θ_j are node bias values.

Example - Data

Obs.	*Fat Score*	*Salt Score*	*Acceptance*
1	0.2	0.9	1
2	0.1	0.1	0
3	0.2	0.4	0
4	0.2	0.5	0
5	0.4	0.5	1
6	0.3	0.8	1

Moving Through the Network
穿过网络

The Input Layer 输入层

For input layer, input = output 在输入层，输入等于输出。

- E.g., for record #1:

  Fat input = output = 0.2

  Salt input = output = 0.9

Output of input layer = input into hidden layer 输入层的输出等于隐藏层的输入。

The Hidden Layer 隐藏层

In this example, hidden layer has 3 nodes 隐藏层有3个节点

Each node receives as input the output of all input nodes

每个节点收到输入层所有节点的输出。

Output of each hidden node is a function of the weighted sum of inputs 隐藏层每个节点的输出是输入值的加权求和值。

$$output_j = g(\Theta_j + \sum_{i=1}^{p} w_{ij}\, x_i)$$

The Weights 权重

The weights q (theta) and w are typically initialized to random values in the range -0.05 to +0.05 权重w和偏置值theta通常初始化为-0.05到0.05之间的随机值。

Equivalent to a model with random prediction (in other words, no predictive value) 这等于做了一个随机预测（换句话说，没有预测值）。

These initial weights are used in the first round of training 这些初始权重会用在第一轮的模型训练中。

Output of Node 3, if g is a Logistic Function

如果g是一个logitstic函数，那么节点3的输出是

$$output_j = g\left(\Theta_j + \sum_{i=1}^{p} w_{ij}\, x_i\right)$$

$$output_3 = \frac{1}{1 + e^{-[-0.3+(0.05)(0.2)+(0.01)(0.9)]}} = 0.43$$

Initial Pass of the Network

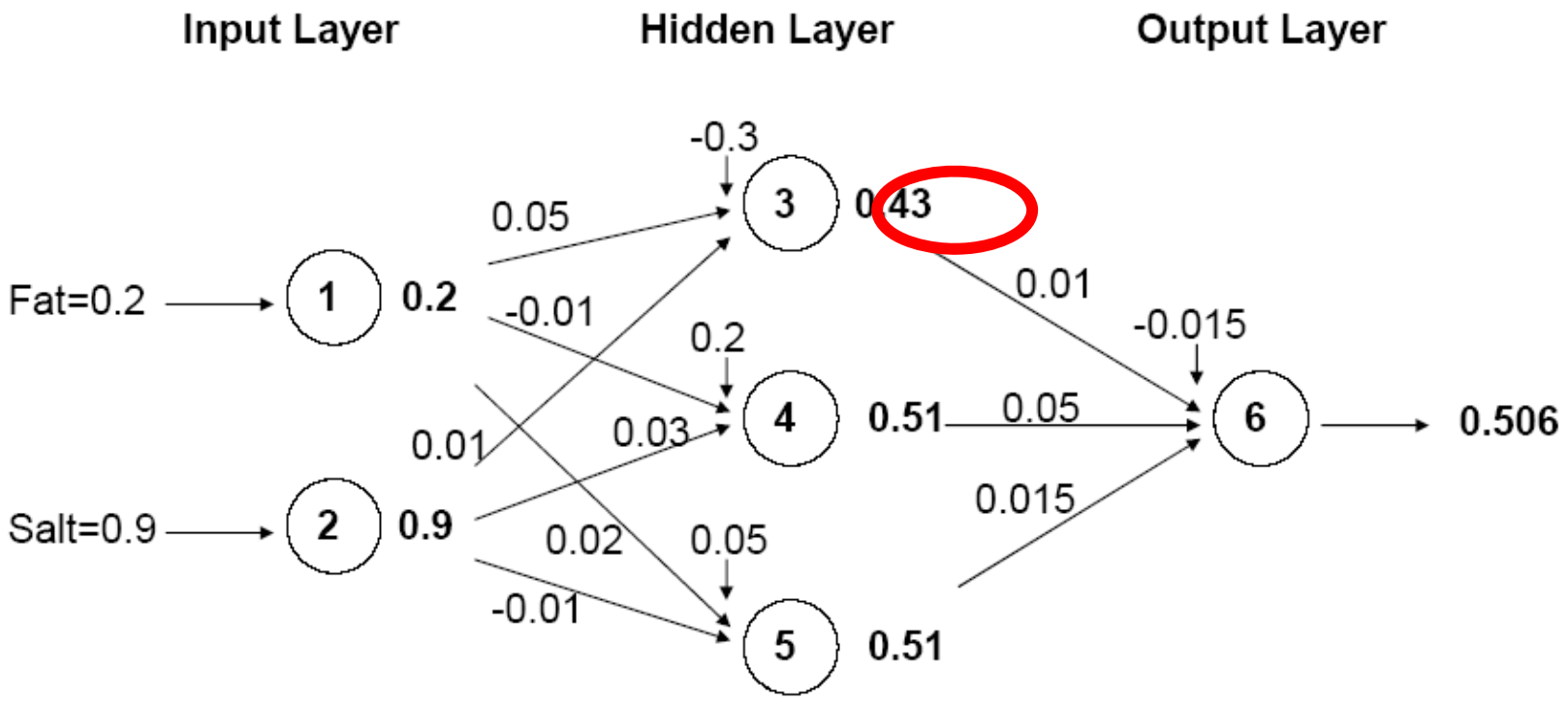


Figure 11.3: Computing node outputs (in boldface type) using the first observation in the tiny example and a logistic function.

Output Layer 输出层

The output of the last hidden layer becomes input for the output layer 最后一个隐藏层的输出成为输出层的输入。

Uses same function as above, i.e. a function g of the weighted average 使用与上面相同的函数（也就是说，加权平均值的某个函数g）。

$$output_6 = \frac{1}{1 + e^{-[-0.015+(0.01)(0.43)+(0.05)(0.507)+(0.015)(0.511)]}} = 0.506$$

The output node

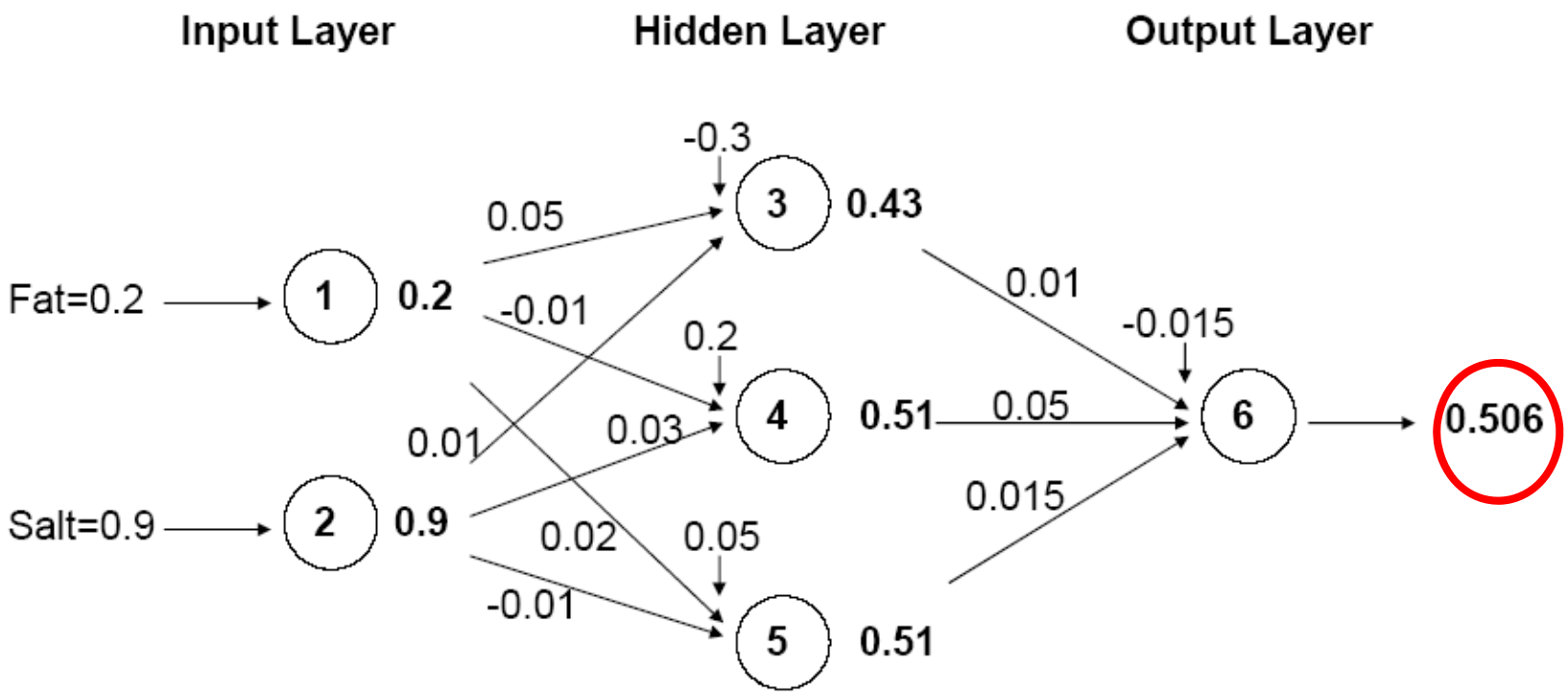


Figure 11.3: Computing node outputs (in boldface type) using the first observation in the tiny example and a logistic function.

Mapping the output to a classification

将输出映射到某个分类。

Output = 0.506 输出等于0.506

If cutoff for a “1” is 0.5, then we classify as “1”
如果把记录分类到1类的截值是0.5，那么我们将该记录分类到1类。

Relation to Linear Regression 与线性回归的关系

A net with a single output node and no hidden layers, where g is the identity function, takes the same form as a linear regression model

线性回归是一个只有一个输出节点且没有隐藏层的神经网络。在该网络中g是恒等函数，形式与线性回归模型使用的函数相同。

$$\hat{y} = \Theta + \sum_{i=1}^{p} w_i x_i$$

Training the Model 训练模型

Preprocessing Steps 步骤

- Scale variables to 0-1 对数值变量进行标准化处理到0-1
- Categorical variables 对于类别型变量的处理如下：
- If equidistant categories, map to equidistant interval points in 0-1 range 如果是等距类别型变量则映射为0-1区间内的等距值。
- Otherwise, create dummy variables 如果不是则建立虚拟变量。
- Transform (e.g., log) skewed variables 对偏斜分布变量可做变换处理（如取log值）。

Initial Pass Through Network
初次通过神经网络

Goal: Find weights that yield best predictions 目标是找到合适的权重值，以得到最好的预测值。

- The process we described above is repeated for all records 我们描述的流程反复用在所有记录上。

- At each record, compare prediction to actual 针对每个记录，比较预测值与真实值。

- Difference is the error for the output node 差别就是输出层节点的错误。

- Error is propagated back and distributed to all the hidden nodes and used to update their weights 错误会反向传播并扩散到所有隐藏层节点并更新它们的权重。

Back Propagation（“back-prop”）反向传播

- Output from output node k: $\hat{y}_k$ 得到输出节点预测值 。

- Error associated with that node: 计算出与该节点相关的错误。

$$err_k = \hat{y}_k(1-\hat{y}_k)(y_k - \hat{y}_k)$$

Note: this is like ordinary error, multiplied by a correction factor

这就类似于普通错误乘以一个校正系数。

Error is Used to Update Weights
错误被用来更新权重值

$$\theta_j^{new} = \theta_j^{old} + l(err_j)$$

$$\mathrm{w}_j^{new} = w_j^{old} + l(err_j)$$

l = constant between 0 and 1, reflects the "learning rate" or "weight decay parameter" 参数*l*是一个0到1之间的常数，代表"学习率"或者"权重衰退参数"。

Case Updating 按单个记录更新

- Weights are updated after each record is run through the network 权重在每个记录通过神经网络之后更新。

- Completion of all records through the network is one *epoch* (also called *sweep* or *iteration*) 所有记录通过网络之后称为一次迭代。

- After one epoch is completed, return to first record and repeat the process 一次迭代结束后，返回到第一个记录继续上述流程。

Batch Updating 批量更新

- All records in the training set are fed to the network before updating takes place 所有训练集的记录通过神经网络之后才开始更新权重。

- In this case, the error used for updating is the sum of all errors from all records 在这种情况下，用来更新权重的错误指标是所有记录预测错误的汇总。

Why It Works 为什么它有效?

- Big errors lead to big changes in weights 大的错误导致大的权重调整。

- Small errors leave weights relatively unchanged 小的错误使权重调整很小或者不调整。

- Over thousands of updates, a given weight keeps changing until the error associated with that weight is negligible, at which point weights change little 上千次调整之后，一个给定的权重不断调整直到与之相关的错误可以忽略不计，至此该权重就变化很小了。

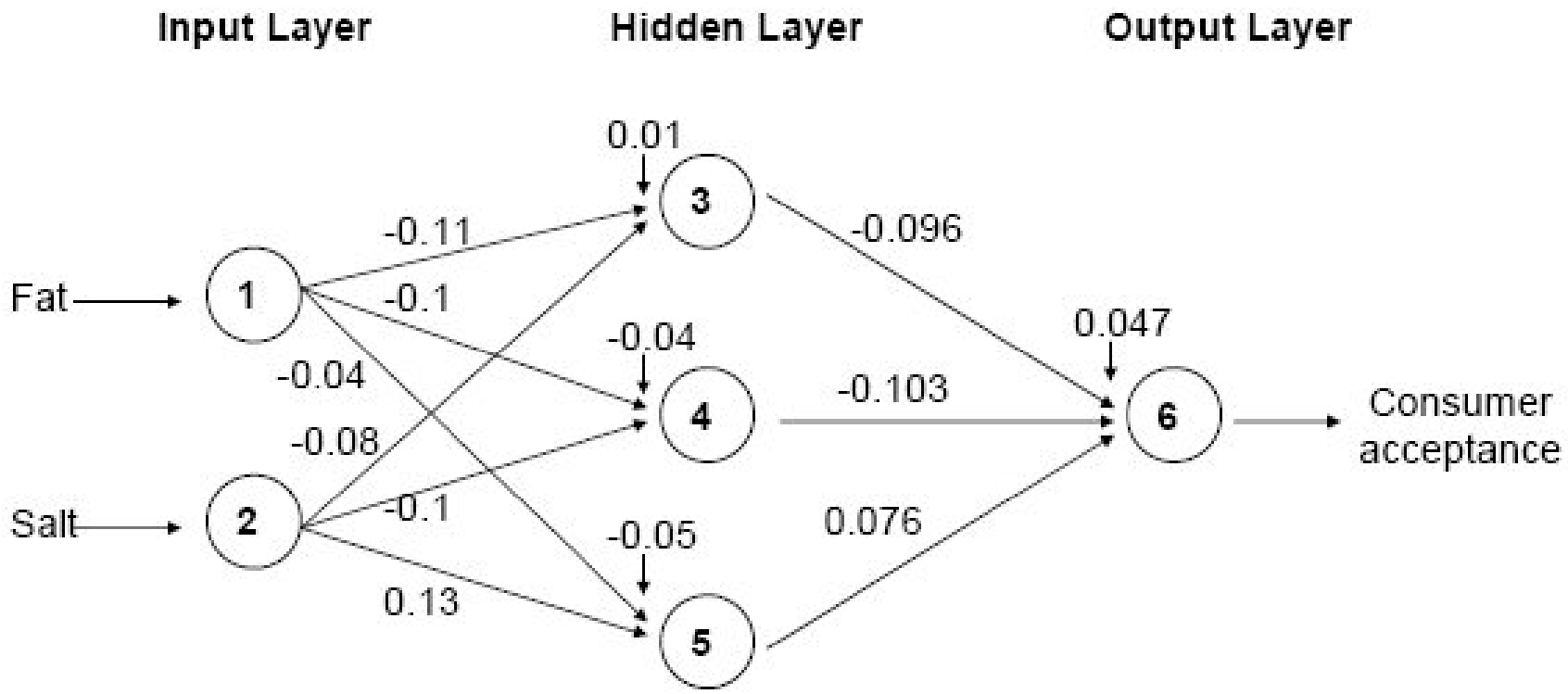
Common Criteria to Stop the Updating
通常停止更新的准则

- When weights change very little from one iteration to the next 当迭代后的权重修正与上次相比非常小。

- When the misclassification rate reaches a required threshold 当错误分类率达到一个预设的门槛。

- When a limit on runs is reached 当迭代次数达到一个预置的上限。

Fat/Salt Example: Final Weights



XLMiner Output: Final Weights

Inter-layer connections weights

	Input Layer		
Hidden Layer # 1	fat	salt	Bias Node
Node # 1	-0.110424	-0.0800683	0.011531
Node # 2	-0.10581	-0.10347	-0.0447816
Node # 3	-0.0400986	0.128012	-0.0534663

	Hidden Layer # 1			
Output Layer	Node # 1	Node # 2	Node # 3	Bias Node
1	-0.0964131	-0.1029	0.0763853	0.0470577
0	-0.00586047	0.100234	-0.0960382	0.0130296

Note: XLMiner uses two output nodes (P[1] and P[0]); diagrams show just one output node (P[1])

XLMiner: Final Classifications

Row Id.	Predicted Class	Actual Class	Prob. for 1 (success)	fat	salt
1	1	1	0.498658971	0.2	0.9
2	1	0	0.497477278	0.1	0.1
3	1	0	0.497954285	0.2	0.4
4	1	0	0.498202273	0.4	0.5
5	1	1	0.49800783	0.3	0.4
6	1	1	0.498571499	0.3	0.8

Avoiding Overfitting 避免过度拟合

With sufficient iterations, neural net can easily overfit the data 进行足够多的迭代之后，人工神经网络非常容易出现过度拟合现象。

To avoid overfitting: 为了避免过度拟合

- Track error in validation data 追踪验证数据集中的错误。
- Limit iterations 限制迭代次数。
- Limit complexity of network 限制网络复杂度。

User Inputs 需要用户手工输入的参数

Specify Network Architecture
定义网络结构

Number of hidden layers 隐藏层的个数

- Most popular – one hidden layer 绝大多数是1个。

Number of nodes in hidden layer(s) 隐藏层节点数

- More nodes capture complexity, but increase chances of overfit 更多的节点可以捕捉到（数据相互关系的）复杂度。但更容易造成过度拟合。

Number of output nodes 输出节点数

- For classification, one node per class (in binary case can also use one) 对于分类型变量，每个节点对应一个分类（二元分类变量则使用1个节点即可）。
- For numerical prediction use one 数值型预测变量使用1个节点。

Network Architecture, cont. 网络结构

“Learning Rate” (l) 学习率 l

- Low values “downweight” the new information from errors at each iteration 学习率取值低就会降低从每次迭代错误中吸取新信息的能力。
- This slows learning, but reduces tendency to overfit to local structure 这导致学习过程变慢，但同时也降低了过度拟合局部数据结构的趋势。

“Momentum” 动量

- High values keep weights changing in same direction as previous iteration 动量项取值高可以让权重的变化保持与上次迭代相同的方向。
- Likewise, this helps avoid overfitting to local structure, but also slows learning 类似的，这有助于避免对局部数据结构的过度拟合，但降低的学习速度。

Automation 自动处理

- Some software automates the optimal selection of input parameters 有的软件可以自动选择最佳输入参数。

- XLMiner automates # of hidden layers and number of nodes

- XLMiner可以自动处理隐藏层数目和节点数目。

Advantages 优势

- Good predictive ability 良好的预测能力
- Can capture complex relationships 可以捕捉复杂的关系
- No need to specify a model 无需事先定义一个模型

Disadvantages 劣势

- Considered a “black box” prediction machine, with no insight into relationships between predictors and outcome 被视为一个黑箱预测机器，不能清楚解释预测因子与结果之间的关系。

- No variable-selection mechanism, so you have to exercise care in selecting variables 没有变量选择的机制，所以你不得不在使用人工神经网络之前小心选择变量。

- Heavy computational requirements if there are many variables (additional variables dramatically increase the number of weights to calculate) 如果有很多预测变量，对计算量要求很高。新增变量会大大增加需要计算的权重值。

Summary 总结

- Neural networks can be used for classification and prediction 神经网络可以用于分类和预测。

- Can capture a very flexible/complicated relationship between the outcome and a set of predictors 可以捕捉到非常复杂/灵活的预测因子与结果之间的关系。

- The network “learns” and updates its model iteratively as more data are fed into it 当更多的数据被提供给网络，它可以不断学习并更新自身的模型。

- Major danger: overfitting 主要的风险：过度拟合

- Requires large amounts of data 需要使用大量数据进行训练。

- Good predictive performance, yet “black box” in nature 预测表现佳，但性质上是一个黑箱。