### 1神经网络基本原理

神经网络模型结构示意图如图3-3（a），作为由M-P神经元模型构成的全连接神经网络，其中的每一个神经元j都接受来自上一层所有神经元的输入信号，进行加权整合后与阈值θj进行比较，并利用激活函数f进行激活后传递给下一层的所有神经元，即：

其中wij为前一层神经元i与当前神经元j间连接权重。把许多个M-P神经元模型按照有层次的方式连接起来，就构成了神经网络模型，或称多层感知机。给定训练数据集,如图，输入变量x有d个分量，输出变量为标量y。因此即输入层有d个神经元，而神经网络的最后一层，即输出层有1个神经元。

wop

E

wph

b

**f**

**f**

a

x

op

oh=

y

(a)神经网络模型结构

(b)误差反向传播算法示意图

…

…

w1j

w2j

wij

wnj

…

…

…

x1

x2

xd

y

…

图3-3 神经网络模型示意图

神经网络模型的训练目标是使模型的输出能准确预测数据集输出变量。为此，设定模型训练的目标函数以衡量模型预测误差，对于回归问题，常用的目标函数包括均方差函数（MSE）：

其中为模型预测值，为了使目标函数尽量减小，从而实现一定精度的预测，神经网络模型在训练中，应用基于梯度下降算法的误差反向传播算法（Back Propagation Algorithm）调整整个神经网络的权重与阈值,如图3-3（b）。假设现在需要在神经网络中使用梯度下降法更新权重wop时，首先需要计算梯度，为此应用“链式法则”，按照信号传播路径的反方向分解梯度：

其中分别为输出层神经元的输入信号和第二层神经元p的输入。由于：

因此：

在选用均方差作为目标函数时,对于任一数据yi：

于是，利用式（3-3）至式（3-8），可以计算出模型中目标函数关于权重wop的梯度。同理能够得到任一权重和阈值的梯度。进而按照式（3-9）（3-10）进行参数更新。误差反向传播算法过程见图3-4：

|  |
| --- |
| **输入**：训练集；学习率 |
| **步骤**：初始化权重与阈值 |
| repeat  for all do  神经网络前向传播，计算当前样本的输出；  计算出神经网络所有权重和阈值的梯度；  更新权重和阈值；  end for  until 达到停止条件  **输出**：权重和阈值确定的神经网络 |

图3-4 误差反向传播算法

仅仅按照基本BP算法，神经网络的训练过程中目标函数下降速度慢，且最终预测精度较低。为了提高模型的预测能力，神经网络还需要应用一系列学习技巧来提高在数据集上的表现。

首先，应用了mini-batch学习法。如图3-1（b）所示误差反向传播算法，在训练过程中，每次参数更新仅仅依据单个数据在神经网络上的计算结果，容易受到噪声的影响。另一种方法是每次训练都将所有训练数据的损失函数总和作为学习指标，此时损失函数变为：

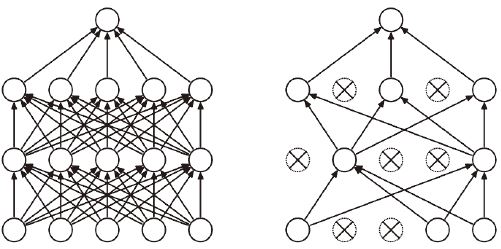
如果数据量过大，例如达到数量级，每次更新都计算所有数据的损失函数总和会花费太长计算时间。本课题采用mini-batch学习法。每次从训练数据中随机选取固定数量的数据进行学习，既减小了数据噪声对学习过程稳定性的影响，也避免了单次学习数据过多导致训练时间成本升高。

其次，需要改进参数更新机制。使用式（3-9）、（3-10）进行参数更新的方法被称为随机梯度下降（SGD）算法。该方法的特点是，采用固定的学习率，如果学习率过小（如0.0001），模型收敛速度会十分慢，但如果学习率过大（如1），在模型学习的后期，参数梯度较小时会发生目标函数震荡，目标函数无法进一步下降。因此，改进后的参数更新方法应使得学习率在学习的初始较大，使得损失函数迅速下降；而在学习过程的后期，梯度较小时学习率应当减小，避免目标函数发生震荡，使得目标函数尽可能达到最低。本课题使用RMSProp方法进行训练，算法如图3-5：

|  |
| --- |
| **输入**：学习率；衰减速率；小常数（一般设为）；初始参数θ； |
| **步骤**：初始化梯度积累变量r=0 |
| repeat  从训练集中采包含n个样本的小批量。  计算梯度：  积累平方梯度：；  计算参数更新：（逐元素应用）；  应用更新：  until 达到停止条件 |

图3-5 RMSProp算法

再次，使用了Dropout层减小过拟合。过拟合是机器学习的常见问题，即模型在训练数据上预测精度较高，但无法良好预测测试集数据。在两种情况下容易发生过拟合[10]：（1）模型拥有大量参数，例如隐藏层数过多，神经元数量过多，模型表现能力强。（2）训练数据少，模型难以学习到具有普适性的规律。应用了Dropout方法的神经网络会在训练过程中随机删除网络中部分神经元，使得这些神经元不再传递信号（如图3-6），但保留这些神经元的参数。随后在测试阶段恢复已删除神经元及其参数。



(a)Dropout前神经网络

(b)Dropout后神经网络

图3-6 Dropout原理示意图[10]

最后，在开始训练前，采用Kaiming初始化方法初始化网络的所有权重。并将所有阈值初始化为0。

### 2决策树模型基本原理

本课题使用的模型3与模型4，即随机森林和XGBoost都以决策树为基学习器。决策树的基本原理是信息增益理论。决策树由根结点，内部结点和叶结点构成，如图3-7（a）。每个数据由根结点进入决策树，沿图中结点间的连接线依次传向下一个结点，决策树的每一个内部结点包含一个判断条件，并依据当前数据的特征向量进行条件判断以决定数据接下来进入哪一个结点，最终进入特定的叶结点，而每一个叶结点都对应一个输出值。

(a)决策树模型示意图

根结点

叶结点

子结点

(x(1),x(2),y)

x(1)=a

x(2)=b

x(1)=c

R3

R3

R4

R1

R2

R1

R3

R4

x(1)

x(2)

a

c

b

(b)特征空间划分示意图

图3-7 决策树模型原理

使用不同的算法，决策树可以用于分类任务或回归任务，经典的算法为CART算法,利用该算法进行决策树模型的训练，所生成的决策树是二叉树结构。假设给定数据集为，为数据总量。,L为特征个数，即输入变量的分量个数。决策时的每一个结点的产生实际上都是对特征空间进行划分，如图3-7（b），而每次划分都需要选择最优特征及划分点，假设选择特征j作为划分特征,并以特征j的某个值s作为划分点，则可以将特则空间划分为两个部分：。为了寻找最佳的特征j和划分点s，需要对下式求解,从而使划分的两个区域平方误差和最小：

是划分得到的两个区域分别对应的输出值，用以下两式计算：

N1，N2分别为被划分入区域R1,R2的数据量。然后对每个划分后的区域重复上述过程，达到停止条件后获得最终决策树。算法步骤如图3-8：

|  |
| --- |
| **输入**：训练集;  属性集; |
| **步骤**：建立结点node；  Repeat  求解式（3-14），选择最佳划分特征j与划分点s；  用选定的特征j和切分点s划分区域并按式(3-15)(3-16)决定相应输出；  Until 满足停止条件。  **输出**：以node为结点的回归树。 |

图3-8 决策树算法

### 2.2决策树模型集成

### 2.2.1随机森林

本课题所使用的第三个集成学习模型，随机森林模型是按照Bagging策略结合多棵决策树的集成学习模型。相比于单棵决策树，具有抗噪声能力强，不易陷入过拟合的优点。

仍以流变应力预测认为为例，设使用了K棵决策树进行集成，设决策树的集合为，在划分好训练集与测试集后，为了让不同基学习器学习到有所不同的规律，一方面使得每个基学习器的训练数据不同，即分别使用Bootstrap采样法从训练集中选取部分样本作为单棵决策树的训练样本。另一方面，对于每棵决策树，只选取一部分特征参与训练，按照一般的经验，选取特征的数量为，选取特征方法为随机选取。这样一来，每个学习器的训练数据和参与决策的特征都有所不同，并最终对所有学习器的预测结果直接平均。

### 2.2.2 XGBoost

本课题所使用的模型4，XGBoost采用的集成学习策略是Boosting策略。首先将一棵树定义如下：

其中w是一个向量，向量中每个元素为一个分数，并与一个特定叶结点对应。q是一个函数，将每个数据与相应的叶结点对应。T是叶结点的数量。之所以如此定义决策树，是为了方便定义接下来的模型复杂度：

整个模型可以表述为：

其中是任意数据的预测值，K是基学习器数量，f是包含所有可能的决策树的泛函空间中的一颗决策树。

接下来定义总体目标函数：

其中n为数据总量。l为衡量预测值与真实值间差别的损失函数，例如均方差函数（式（3-2））。

假定已经使用XGBoost算法生成了t-1棵树，接下来需要生成第t棵树。Boosting集成策略在应用时，基学习器间的学习不是独立的，每一个新生成的学习器都受到之前已经生成的学习器影响。令第t棵树生成后的目标函数为：

代入（3-17）,（3-18），得：

将目标函数式进一步压缩得：

Gj, Hj分别为每个叶结点的一阶和二阶导数的和。是分配给第j个叶结点的所有数据的索引。

XGBoost模型的学习目标是使目标函数达到最低，为此需求解方程：

解得：

将式（3-28）代入（3-25），得：

即目标函数的最终形式。

接下来需要确定单棵树的生成步骤。XGBoost采用的是二叉树模型，在每一次分裂结点时，对所有叶结点都进行一次分裂。而判断是否需要分裂的标准是,增益大于0。定义增益如下：

等式从左到右各项的含义分别为分裂后新左叶结点上的得分、分裂后新右叶结点上的得分，原始叶结点上的得分以及最右边的正则化项。如果不包含正则化项的增益小于，那么gain小于0，不进行新的分裂。正则化项的存在抑制了树模型深度过大，即抑制了模型的过拟合。

在判断需要分裂的前提下，接下来需要选择特定特征进行分裂。XGBoost采用并行方法，尝试对所有特征进行分裂，计算分裂后的增益，并最终将能够获得最大增益的特征作为最佳特征。

对每个特征进行分裂时，都需要选择分裂点位，即选定特征的取值。类似于特征选择，选择分裂点时采用的是全局扫描法，即将数据训练集内该特征的所有分裂点都进行尝试，选择增益最大的取值作为最佳特征的划分点。