一种深度梯度提升回归预测模型

曲文龙[1] 陈笑屹[1] 李一漪[1] 汪慎文[1,2]
[1](河北地质大学信息工程学院 河北 石家庄 050031)
[2](中国科学院自动化研究所 北京 100190)

摘 要 浅层学习模型对复杂函数表示能力有限,从而导致泛化能力受到制约。针对此问题,结合深度学习和集成学习思想提出一种基于深度梯度提升的回归预测模型。该模型在输入层对原始特征进行特征子集提取,训练生成子空间基学习器;隐藏层通过构建多层级联结构,逐层融合子空间特征与原始特征从而实现逐层表征学习,并根据相邻层学习变化率自适应学习层数;输出层中使用学习法结合策略对样本进行最终预测。采用并行化方式对各层学习器进行训练以提高模型运行效率。在 UCI 公开数据集上进行实验验证,结果表明:相比现有集成预测方法,该模型具有更高的预测精度和运行效率。

关键词 梯度提升 深度学习 集成学习 回归预测

中图分类号 TP181 **文献标志码** A DOI:10.3969/j.issn.1000-386x.2020.09.032

A REGRESSION PREDICTION MODEL OF DEPTH GRADIENT BOOSTING

Qu Wenlong[1] Chen Xiaoyi[1] Li Yiyi[1] Wang Shenwen[1,2]
[1](*School of Information Engineering*, *Hebei GEO University*, *Shijiazhuang 050031*, *Hebei*, *China*)
[2](*Institute of Automation*, *Chinese Academy of Sciences*, *Beijing 100190*, *China*)

Abstract Shallow learning model has limited ability to express complex functions, so its generalization ability is restricted. Aiming at this problem, a regression prediction model based on depth gradient boosting is proposed by combining deep learning and ensemble learning. The model extracted the original feature subset in the input layer and trained the subspace based learner. The hidden layer merged the subspace features and the original features layer by layer by adopting a multi-layer cascade structure, so as to realize the layer-by-layer representation learning. In addition, it could also learn the number of adaptive learning layers according to the rate of change of adjacent layers. In the output layer, strategies of learning method were used to make the final prediction of the samples. The parallel method was adopted to train the learners to improve the running efficiency of the model. Experimental results on UCI open data set show that our model has higher prediction accuracy and efficiency than the existing ensemble prediction methods.

Keywords Gradient boosting Deep learning Ensemble learning Regression prediction

0 引 言

回归模型是通过对统计关系的定量描述,以实测和统计为依据,确定各相关变量间的拟合关系的一种数学模型[1]。回归问题是机器学习主要应用方向之一,Friedman[2] 指出回归问题实际上是函数空间的优化问题,目的是求出因变量关于自变量的函数,使损失函数的期望最小。回归预测广泛应用于国民经济与社会生产等领域。

统计学习理论是一种研究小样本机器学习规律的理论,它从实际出发,提取数据特征,构建概率统计模型进行数据的分析与预测[3]。在统计学习的建模过程中,建立输入变量与输出变量之间的函数关系,最常见

收稿日期:2019-06-04。国家自然科学基金项目(61402481);河北省重点研发计划项目(18212005);河北省教育厅自然科学基金重点项目(ZD2018083)。**曲文龙**,教授,主研领域:数据挖掘,机器学习,并行计算。**陈笑屹**,硕士生。**李一漪**,硕士生。**汪慎文**,副教授。

的统计学习模型基于线性回归模型,这类方法原理简单,便于操作,但是十分依赖于以往经验,针对复杂问题容易发生精度偏差[4]。机器学习是人工智能的重要组成部分,传统机器学习回归方法有支持向量机回归[6-7]、神经网络回归[8-9]和贝叶斯回归[10]等,目前已应用于疾病预测、环境监测和股票预测[11]等各个领域。然而这些单一方法存在局限性且稳定性较差、精度不高,无法应对大数据量的精准分析等。集成学习通过整合多个学习器来完成学习任务,以降低泛化误差。其优势在于将多个个体学习器合并,得到更加合理的边界,以降低整体的错误率、提高模型性能[12]。集成学习的方法分为 Bagging[13]、Boosting[14]和 Stacking[15]三类,其代表算法有随机森林[16]、GBDT[17]和 XGBoost[18]等。

传统的浅层学习模型对特征的依赖性强,表达能力有限。深度学习算法对于复杂问题具有更强的处理能力,它的成功主要依赖于深度结构,通过多层级联结构将原始特征组合抽象出高级特征[19-20]。深度学习通过构建具有多个隐藏层的网络模型,使用海量数据进行训练,实现自动提取更复杂、更有用的特征,从而具有远超浅层网络的表达能力[21-22]。但深度神经网络的超参数多,参数调节较为困难;需要一定数量的训练样本,少量的训练样本会导致欠拟合的情况;为了保证运行速度,复杂的神经网络结构需要有昂贵的硬件支持。将集成学习方法与深度学习思想相结合,提供了一个有前途的研究途径。Qi 等[23]提出的深度支持向量机(DeepSVM)模型是一种新的深度结构,利用 Ex-Adaboost 学习策略来选择错误率较低、多样性较高的 SVM 学习器,之后经过每一层生成新的训练数据,完成最终的分类或回归任务。Zhou 等[24]提出了深度森林模型 gcForest(多粒度级联森林,Multi-Grained Cascade Forest)。它是生成一个深度树集成方法,使用级联结构让 gcForest 做表征学习,当输入为高维时,通过多粒度扫描,其表征学习能力还能得到进一步的提升,其优势是可以处理小数据量,且超参数少、鲁棒性高。

本文提出一种深度梯度提升模型,使用梯度提升回归树作为个体学习器,采用多层级联结构进行逐层表征学习。以并行化的方法训练个体学习器,进一步提升学习效率。模型可根据数据自动适应学习层数,拟合风险小,超参数少,运行速度快。

1　集成学习算法

1.1　集成学习元算法

集成学习建立一组学习器,基于某种规则将各个学习器的结果整合,从而获得比单个学习器更好的学习效果,提高预测精度并降低过拟合风险。1990 年 Schapire[25]证明了在概率近似正确(Probably Approximately Correct,PAC)学习框架下,一个概念是强可学习的充分必要条件是这个概念是弱可学习的,从而为集成学习奠定了理论基础。集成学习元算法可分为 Bagging、Boosting 和 Stacking 三类。Bagging 方法通过对训练样本、特征属性集随机采样,学习多个独立模型,对学习结果做平均或投票做出最终的预测,可减小模型方差[13]。Boosting 是一个逐步加强的迭代方法,每次学习基于前面模型的训练误差,强调偏差大(错误)的样本,对样本的分布进行调整,最后对多个弱学习器预测值加权组合,可减小模型偏差[14]。Stacking 的基本思想是训练一个基本分类器池,然后使用另一个分类器来组合它们的预测,以降低泛化误差[15]。

1.2　梯度提升回归树

梯度提升回归树(Gradient Boosting Regression Tree, GBRT)是 FrideMan 在 2000 年提出的一种 Boosting 方法[26],基学习器为 CART 回归树。每次迭代中,利用当前模型中损失函数的负梯度值作为提升树算法中的残差的近似值,进而拟合一棵回归树,并叠加得到新模型。对于平方损失函数负梯度就是残差,对于一般损失函数,负梯度可视为残差的近似值。

算法 1　梯度提升回归树(GBRT)算法

输入:训练样本集 $\{(x_i,y_i)\}_{i=1}^{N}$,最大迭代次数 M,可微损失函数 $L(y,F(x))$。

输出:强学习器 $y=F_M(x)$。

初始化模型为常数:

$$F_0(x)=\arg\min_{\gamma}\sum_{i=1}^{N}L(y_i,\gamma) \tag{1}$$

for $m=1$ to M:

计算伪梯度:

$$r_{im}=-\left[\frac{\partial L(y_i,F(x_i))}{\partial F(x_i)}\right]_{F(x)=F_{m-1}(x)} \quad i=1,2,\cdots,N \tag{2}$$

使用 $\{(x_i,r_{im})\}_{i=1}^{N}$,拟合一棵伪梯度回归树 $h_M(x)$。

计算回归树 $h_M(x)$ 重系数 γ_m:

$$\gamma_m=\text{argmin}_{\gamma}\sum_{i=1}^{N}L(y_i,F_{m-1}(x_i)+\gamma h_m(x_i)) \tag{3}$$

更新模型:

$$F_m(x)=F_{m-1}(x)+v\gamma h_m(x) \tag{4}$$

// $v\in(0,1)$ 为收敛参数

end for

输出 $F_M(x)$

GBRT 组合多个弱学习器可以防止过拟合,具有非线性变换处理能力,不需做特征变换,其缺点是基学习器通过迭代顺序生成,难以实现并行化。

1.3 极限梯度提升树

极限梯度提升树(eXtreme Gradient Boosting,Xgboost)是由 Chen 等[18]提出的一种正则化的和可伸缩的梯度提升算法。传统的梯度提升算法优化时只用到损失函数一阶导数信息,Xgboost 对损失函数进行了二阶泰勒展开,同时利用了一阶和二阶导数,并且可自定义损失函数。Xgboost 在损失函数里加入了正则项,用于控制模型的复杂度,综合权衡模型的偏差和方差,简化模型并且防止过拟合。算法具有良好的预测性能,可处理空缺和稀疏数据、支持特征粒度的多核和分布式多处理器并行计算。

2 深度梯度提升模型

2.1 模型结构

深度梯度提升模型使用级联结构的深度集成方法对特征向量进行逐层表征学习,分为输入层、隐层和输出层。

输入层(L_1)包括若干学习器(R_{11}, $R_{12},\cdots,R_{1m}$)进行初级特征学习,每个学习器使用随机子空间方法随机选择相同大小的不同特征组合的子空间作为输入。隐层中含有隐层学习器进行高层特征抽象。为保持数据集原始特征信息,第一层隐层(L_2)的输入为原始特征和输入层若干学习器的输出。从第二层隐层开始(L_3),每一层的输入包含原始数据集中的所有特征和所有隐层学习器的输出作为下一层隐层学习器的输入。根据学习结果,隐层层数自适应确定,当上一层的预测结果矩阵与当前层预测结果矩阵的差值绝对值矩阵中每一项值的平均值小于容忍度 ε 时停止增加层数。输出层(L_n)采用学习器,对最后隐层输出和原始输入特征进行融合预测,得到最终预测输出。其级联结构如图 1 所示。

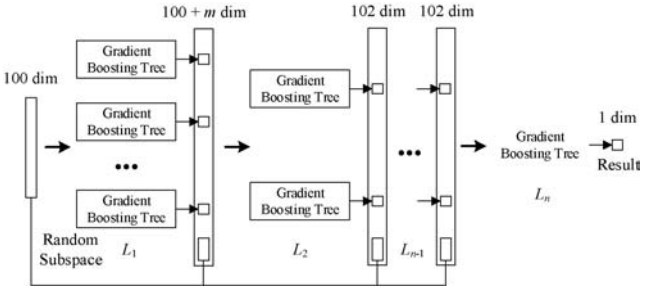

图 1 深度梯度提升模型级联结构

假设有 100 维输入特征,输入层结构将在原始的训练集中使用随机特征子空间的方法训练 m 个学习器并对所给定样本进行预测,形成 m 维的数据集合,与原始 100 维输入特征按列进行合并,形成 $100+m$ 维的数据集合作为首层隐层(L_2)中所有学习器(R_{21},R_{22})的输入。

从隐层的第二层(L_3)开始,隐层中的输入为上一隐层中 2 个学习器的输出的预测结果和原始 100 维输入特征按列进行合并的数据集合,共 102 维,作为下一隐层学习器的输入。以此类推,直到当前隐层与上一隐层的预测结果的每一项差值的绝对值的平均数小于容忍度 ε 时,则停止增加隐层。

输出层的输入为最后一层隐层中 2 个学习器的输出预测结果向量(2 维)与原始的 100 维输入特征按列进行合并的数据集合,共 102 维,输出的最终预测结果为 1 维向量。

2.2 随机子空间学习

输入层将原始的输入特征进行特征提取。对于 d 维属性,随机抽取不大于$\sqrt{d}$的最大整数作为一次选取中的属性数(参照 Breiman 在随机森林中的特征子集选择方法),然后用已选取的属性集合训练一个学习器。将上述步骤执行多次,得到了包含若干个学习器节点的节点集合,即模型的输入层(L_1)。每个节点的输出为一个预测值向量,将多个节点输出的预测值按列进行合并就形成了预测值向量集合。

输入层节点以随机子空间方式进行特征提取,可能存在部分节点因差异性和互补性较低,进而导致输入层性能下降的问题。本模型以平均相似度作为衡量标准,去除输入层中较为相似的节点,使得保留下的节点尽可能具有较高的差异性和互补性,以此提高输入层的性能和运行效率[27]。设输入层节点数为 k,则输入层第 i 个节点与输入层其他节点的平均相似度(Average Similarity,AS)的计算公式为:

$$AS_i = \frac{1}{k-1}\sum_{\theta_i \neq \theta_j}\rho(\theta_i,\theta_j) \tag{5}$$

式中:θ_i 为第 i 个节点的输出预测结果,$\rho(\theta_i,\theta_j)$ 为第 i 个节点的预测结果与第 j 个节点的预测结果的相关系数。

输入层节点的总体平均相似度(Overall Average Similarity,OAS)计算公式为:

$$OAS = \frac{1}{k}\sum_{i=1}^{k} AS_i \tag{6}$$

将大于总体平均相似度的节点的输出数据与原始的输入训练集按列进行堆叠,作为隐层第一层(L_2)的输入。输入层的随机子空间学习算法如下:

算法 2 随机子空间学习算法

输入:训练样本 D_{x_train},训练标签 D_{y_train},测试样本 D_{x_test},特征数 d,输入层学习器 $R_1 = \{R_{11},R_{12},\cdots,R_{1n}\}$,节点数 n。

输出:隐层 x _train',隐层训练标签 y _train',隐层测试样本 x _

test'。

$r_{train} = \{\varnothing\}$　　//训练样本的预测结果集合(按列合并)

$r_{test} = \{\varnothing\}$　　//测试样本的预测结果集合(按列合并)

for $i = 1$ to n do

$F = \text{selectRandom}(D_{x\_train}, \sqrt{d})$

//为训练数据随机选取 $\sqrt{d}$ 个特征子集

$D'_{x\_train} = \text{selectRows}(D_{x\_train}, F)$

//为节点 i 生成 $\sqrt{d}$ 维特征子空间训练样本

$D'_{x\_test} = \text{selectRows}(D_{x\_test}, F)$

//为节点 i 生成 $\sqrt{d}$ 维特征子空间训练样本

$D'_{y\_train} = D_{y\_train}$　　//保持训练标签不变

$h_i \leftarrow R_{1i}(D'_{x\_train}, D'_{y\_train})$

//训练生成第 i 个梯度提升树 h_i

if $AS_i \leqslant OAS$ do

//按式(5)和式(6)计算第 i 个梯度提升树 h_i 的平均相似

// 度 AS_i 是否小于或等于总体的平均相似度 OAS

$r_{train} = r_{train} \cup h_i(D'_{x\_train})$

//将 h_i 对 D'_{x_train} 的预测结果按列合并到 r_{train}

$r_{test} \leftarrow r_{test} \cup h_i(D'_{x\_test})$

//将 h_i 对 D'_{x_test} 的预测结果按列合并到 r_{test}

end if

end for

$x\_train' \leftarrow r_{train} \cup D_{x\_train}$ // 原始 d 维特征与 m 维预测结果

// 按列合并,形成 $d + m$ 维训练数据样本

$y\_train' \leftarrow D_{y\_train}$　　//保持训练标签不变

$x\_test' \leftarrow r_{test} \cup D_{x\_test}$ // 原始 d 维特征与 m 维预测结果按

// 列合并,形成 $d + m$ 维测试数据样本

输入层中每一学习器的输入是对原始数据随机抽取的特征子集,各节点采用了随机子空间的方法进行抽取,有利于选择更适当的预测特征组合。输入层的输出结果的每一维度是基于不同特征组合所得出的预测结果,其保持了个体学习器的差异性,作为隐层的输入有利于提高模型的泛化能力,将其与原始特征组合后作为隐层输入,在对原始信息进行高维抽象的基础上,保留了原始样本的信息,避免了隐层决策信息丢失。

2.3　多层表征学习

隐层是由若干个学习器节点组成的级联网状结构,用于对输入层输出的预测值和原始特征组合进行高层特征学习。隐层的第一层根据输入层的输出和原始特征集按列进行合并作为输入;第二层根据上一隐层的输出和原始的特征集按列进行合并作为输入。为降低过拟合风险,隐层层数依据当前隐层预测结果与上一层预测结果变化率均值 c 和容忍值 ε 自动调整,容忍值 ε 为学习结果变化显著性参数,取值为可容忍预测误差下限。当 c 大于容忍值 ε 时继续学习,当 $c < \varepsilon$ 时停止训练。

设隐层节点数为 k,样本数为 n,a_{ij} 表示当前隐层中第 i 个样本在第 j 个学习器的预测值,b_{ij} 表示上一隐层中第 i 个样本在第 j 个学习器的预测值,c 值计算公式如下:

$$c = \frac{1}{kn}\sum_{i=1}^{n}\sum_{j=1}^{k}\left|\frac{a_{ij} - b_{ij}}{a_{ij}}\right| \tag{7}$$

隐层表征学习算法如下:

算法 3　多层表征学习算法

输入:隐层训练样本 D_{x_train},训练标签 D_{y_train},测试样本 D_{x_test},第 i 隐层学习器 $\{R_{i1}, R_{i2}\}$ 容忍度 ε,原始训练集,原始测试集。

输出:输出层训练集 D_{x_train}、训练标签 D_{y_train}、输出层测试集 D_{x_test}、隐层层数 i。

$i \leftarrow 1$　　//隐层层数计数变量

$c \leftarrow \infty$　　//相邻层预测累计绝对差值

do while ($c > \varepsilon$)　　//累计绝对差值 $\leqslant \varepsilon$ 时算法终止

$h_{i1} \leftarrow R_{i1}(D_{x\_train}, D_{y\_train})$;

//训练生成第 i 隐层的第一个梯度提升树 h_{i1}

$h_{i2} \leftarrow R_{i2}(D_{x\_train}, D_{y\_train})$;

//训练生成第 i 隐层的第二个梯度提升树 h_{i1}

$D_{x_train} = h_{i1}(D_{x_train}) \cup h_{i2}(D_{x_train}) \cup$原始训练集;

//对每一训练样本,将本层两个学习器输出与原始训

//练集按列合并组成新的训练样本

$D_{x_test} \leftarrow h_{i1}(D_{x_test}) \cup h_{i2}(D_{x_test}) \cup$原始测试集;

//对每一训练样本,将本层两个学习器输出与原始训

//练集按列合并组成新的训练样本

计算当前层预测结果与上一层预测结果变化率 c;

$i = i + 1$;

end do

隐层中每一层的输出都是对之前层数(含输入层)特征的一个高层特征概括,有利于取得良好的预测结果。使用原有数据与每一层隐层学习器的输出进行合并可以在进行高维特征提取时保持原有数据集信息,防止因数据信息丢失而导致预测结果不准确。隐层的层数确定体现了学习结果的变化情况,防止因过多的隐层导致模型过拟合或过少的隐层导致模型欠拟合。ε 体现了对于当前隐层和上一级隐层差异值的容忍程度,决定了隐层层数的确定规则。当上一级隐层与当前隐层的差值小于 ε 时,说明后续的训练即使继续增加隐层数预测结果的变化仍不太明显,即达到收敛。因此差值小于 ε 时就可停止训练,确定当前隐层为隐层中的最后一层隐层。

2.4 学习法结合策略

当隐层数目确定,即隐层高维特征抽象和提取过程结束,将进行最后的预测。根据学习法的结合策略,同样地,将隐层的输出结果和原始的特征集合按列进行合并作为输出层学习器的输入,输出的是对于原始数据集的预测结果。具体算法如下:

算法 4 学习法结合策略预测算法

输入:隐层输出的新的训练样本 D_{x_train},训练标签 D_{y_train},测试样本 D_{x_test},输出层学习器 R,原始训练集,原始测试集。

输出:预测结果 r。

$h \leftarrow R(D_{x\_train} \cup$ 原始训练集$, D_{y_train})$;

//用隐层输出的新的训练样本与原始训练集按列合并训练得到输出层学习器 h

$r \leftarrow h(D_{x_test} \cup$原始测试集$)$; //用 h 预测原始测试集

输出层使用个体学习器进行最后的预测结果输出,体现了学习法的结合策略,有利于扩大假设空间、降低了陷入局部极值的风险、提高了泛化性能。将之前隐层的输出同时堆叠原始的特征信息、继续保留原始数据信息和高层信息作为最后输出层学习器的输入有利于求解出偏差值更低的预测结果。

2.5 模型分析

深度梯度提升模型较传统 Stacking 集成策略更为复杂,是在梯度提升模型上的再次集成。每一层结构在抽象高维特征的基础上融合原始数据特征,并采用不同的学习器以提高模型的泛化能力。自适应结构可根据数据预测结果自行确定层数,降低了欠拟合或过拟合的风险。输入层、隐层学习可并行化学习,即同一层中的每个节点同时进行训练,提高了模型的运行效率。相比深度神经网络,深度梯度提升树模型的超参数少,训练调参简单。相比传统的集成学习方法,本文所提出的方法可获得更高的预测精度。

3 实 验

3.1 实验设计

设计四个实验验证模型优越性、有效性和参数敏感性。实验一采用 6 个 UCI 数据集,将本文提出的深度梯度提升模型与回归树模型、Xgboost 集成模型、Stacking 结合策略集成模型在不同数据集上进行性能对比分析,以验证所提出模型的优越性。实验二通过在每个数据集上不同输入层节点选择策略的预测评价指标值,分析输入层节点数对预测性能的影响。实验三通过不同隐层数下各数据集的预测评价指标值,分析隐层层数对预测性能的影响,并验证采用适当的容忍度 ε 值可自适应得到最优隐层数目。实验四分别以串行和线程并行方式对输入层进行构建,对比不同构建方式下的运行时间,验证模型并行化实现的有效性。

对比实验中回归树采用经典决策树回归算法;梯度集成学习模型采用 Xgboost 算法;Stacking 结合策略集成学习模型中,首先由多个 Xgboost 分别进行学习,之后由 Xgboost 或 GBRT 对多个 Xgboost 的学习结果进行 Stacking 集成。深度梯度提升模型输入层 L_1 采用多个 Xgboost,隐层 $L_2, L_3, \cdots, L_{n-1}$ 分别采用 1 个 Xgboost 和 1 个 GBRT,输出层 L_n 采用 Xgboost。

3.2 实验环境

实验所用计算机配置为 Intel Core i7 7700HQ (2.80 GHz),16 GB 内存,所运行的软件环境为安装在 Windows 10 上的 Python 3.7.1。

3.3 实验方法

实验中使用 10 折交叉验证的方法对原始样本进行随机划分生成训练集、训练标签和测试集、测试标签,其中训练集占样本的 70%,测试集占 30%。

3.4 评价指标

回归评价指标包括:平均绝对误差(MAE)、均方误差(MSE)、均方根误差(RMSE)、平均绝对百分误差(MAPE)、希尔不等系数(TIC)和 R^2 系数(拟合优度)等。拟合优度 R^2 是回归平方和在总平方和中所占比率,即回归方程所能解释的因变量变异性的百分比,便于对不同回归模型的拟合程度进行比较,其值越接近 1 表明模型对观测值的预测性能越好。实验中选取 R^2 系数作为预测模型评价指标,其计算公式如下:

$$R^2(y,\hat{y}) = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{8}$$

3.5 实验结果分析

3.5.1 算法预测性能比较

实验一将本文提出的模型与传统预测模型进行对比,三种对比模型为:决策树回归模型、Xgboost 集成学习模型和 Stacking 结合策略集成学习模型和深度梯度提升树预测模型。

使用的 6 个 UCI 数据集包括:Air Quality、Bike Sharing、Boston 、Diabetes、PRSA 和 Wine。其中,Wine 数据集中将两个数据集文件合并为一个文件,并新加

一列类型属性,对其加以区分。数据集统计信息如表 1 所示。

表 1　数据集统计信息

序号	数据集	样本数	特征数	处理后特征数
1	Air Quality	9 282	15	15
2	Bike Sharing	8 646	16	16
3	Boston	506	13	13
4	Diabetes	442	10	10
5	PRSA	8 761	12	8
6	Wine	6 498	11	12

超参数是指机器学习模型的参数,深度梯度提升树的超参数包括:输入层节点数,随机子空间特征数,隐层层数。其中:输入层节点数按照经验公式 $d - \lfloor\sqrt{d}\rfloor$ 进行确定(d 为特征数),输入层每次随机选取的属性数为不超过 $\sqrt{d}$ 的最大整数,隐层层数依据容忍值 ε 自适应确定,实验中设置 $\varepsilon = 0.015$。将不同模型在每个数据集中的 R^2 系数指标进行对比,实验中取每个数据集随机划分训练集和测试集时运行 200 次结果的平均值,得到四种模型在不同数据集上的预测结果,如表 2 所示。

表 2　各算法预测性能比较　　%

数据集	决策树	XGBoost	Stacking	深度梯度提升
Air Quality	62.19	77.86	85.68	91.00
Bike Sharing	42.62	67.37	69.97	72.61
Boston	62.38	80.88	77.73	81.23
Diabetes	18.85	20.72	42.76	49.52
PRSA	11.62	29.27	40.59	56.97
Wine	8.64	14.31	15.21	21.42

可以看出,相比 XGBoost、Stacking 集成模型和深度梯度提升模型,经典的决策树回归模型在上述 6 个 UCI 数据集中的 R^2 系数相对较低;Stacking 集成模型的 R^2 系数在多数情况下高于 XGBoost 模型,即 Stacking 集成模型在准确度性能方面较原始 XGBoost 模型有所提升;深度梯度提升树模型在每个 UCI 数据集上的 R^2 系数均高于前三种算法,相比 Stacking 集成模型,在 R^2 系数上平均提高了约 5.94 %。结果表明,在上述 6 个实验数据集中,本文模型的预测性能优于决策树回归算法、XGBoost 和 Stacking 集成学习算法。

3.5.2　输入层节点数对预测性能的影响

实验二为研究输入层节点数对预测性能的影响,取得使得 R^2 系数较高的输入层节点数选择策略,将深度梯度提升模型的输入层节点数选择策略分别设置为 $\lfloor\sqrt{d}\rfloor$、$d - \lfloor\sqrt{d}\rfloor$、$\lfloor \ln d \rfloor$、$d$、$2d$($d$ 为数据集原始特征数)。在不同数据集上使用上述输入层节点选择策略对原始数据集进行回归预测,计算并对比经过 20 次运行后各数据集在各输入层选择策略下的各 R^2 系数平均值,实验结果如表 3 所示。

表 3　输入层节点数对 R^2 系数影响　　%

数据集	$\lceil\sqrt{d}\rceil$	$d-\lceil\sqrt{d}\rceil$	$\lceil \ln d \rceil$	d	$2d$
AirQuality	90.53	91.12	90.53	90.35	90.14
BikeSharing	71.93	72.27	71.55	71.33	71.17
Boston	81.71	82.10	80.88	81.07	81.41
Diabetes	46.01	50.51	43.72	44.43	45.32
PRSA	54.02	57.18	55.10	52.70	55.31
Wine	18.93	19.58	18.14	18.40	19.49

可以看出,对于实验数据集,输入层节点数的设置对于最终数据集预测结果的 R^2 系数存在一定影响。输入层节点数设置为 $\lfloor\sqrt{d}\rfloor \sim 2d$ 之间对预测结果影响不显著;R^2 系数设置为 $d - \lfloor\sqrt{d}\rfloor$ 时,相比其他选择策略预测结果精度更高。

3.5.3　隐层层数对预测性能的影响

实验三分析深度梯度提升模型的隐层层数对预测精度的影响。在其他参数不变的情况下,分析隐层数取值变化时,对不同数据集上的 R^2 系数的影响(容忍度值取 $\varepsilon = 0.015$),运行结果如表 4 所示。

表 4　不同隐层数对 R^2 系数影响　　%

隐层数	Airquality	BikeSharing	Boston	Diabetes	PRSA	Wine
1	88.15	69.29	56.06	43.25	54.10	14.72
2	90.06	69.68	82.04	46.22	48.35	16.75
3	90.26	70.82	72.19	50.80	51.95	20.27
4	90.51	71.52	77.41	48.80	52.16	19.25
5	90.97	69.38	77.26	43.28	52.36	17.53
6	91.02	69.86	70.44	48.42	53.48	16.63
7	91.10	70.64	72.68	49.25	54.11	17.45
8	90.87	70.52	77.55	47.97	57.96	18.31
9	90.76	70.28	80.30	43.86	57.65	17.82
10	89.48	69.98	74.86	46.15	57.12	17.30

可以看出,随着隐层数变化,R^2 系数值也呈现明显变化。对于 Airquality 数据集,隐层数从 1 增加到 7 时 R^2 系数逐渐提高,在第 7 层时 R^2 系数达到最高值:91.10%,从第 8 层开始 R^2 系数开始降低,说明当达到第 7 层时,依据 ε 取值自动终止构建隐层。若继续构建

隐层,则因过拟合而导致预测精度降低;对 BikeSharing 数据集,当隐层数为 4 时 R^2 系数达到最高值 71.52%;对 Boston 数据集,当隐层数为 2 时 R^2 系数达到最高值:82.04%;对 Diabetes 数据集,当隐层数为 3 时 R^2 系数达到最高值:50.80 %;对 PRSA 数据集,当隐层数为 8 时 R^2 系数达到最高值:57.96%;对 Wine 数据集,当隐层数为 3 时 R^2 系数达到最高值:17.44%。综上所述,隐层的数量对于预测精度(R^2 系数)存在影响,隐层的数量需要限制在一定范围内,过少或者过多的隐层数都会降低预测精度。

3.5.4 并行化运行效率比较

实验四为研究不同输入层并行化构建方式对于深度梯度提升模型的运行效率的影响,分别以串行方式(1 线程)和多线程并行方式对输入层节点进行训练。设输入层有 $d-\left|\sqrt{d}\right|$ 个节点,则在 k 线程并行化方式的条件下,每个线程任务中的节点数为 $\left|\frac{d-\left|\sqrt{d}\right|}{k}\right|$ 或 $\left|\frac{d-\left|\sqrt{d}\right|}{k}\right|$。在不同数据集上进行实验,对比不同数据集中串行和并行输入层构建方式的运行时间,其并行线程数不超过 $d-\left|\sqrt{d}\right|$,实验结果如表 5 所示。

表 5 算法并行化运行时间对比 s

数据集	节点数	1 线程	2 线程	4 线程	8 线程
Air Quality	6	1.860 6	0.961 5	**0.648 2**	–
BikeSharing	12	1.821 6	1.029 2	0.599 8	**0.437 4**
Boston	9	0.402 6	**0.281 1**	0.313 2	0.328 0
Diabetes	6	0.264 6	**0.184 8**	0.202 2	–
PRSA	5	0.616 0	0.316 7	**0.199 6**	–
Wine	8	0.972 0	0.555 4	0.338 4	**0.333 7**

可以看出,对各数据集输入层串行构建执行时,运行效率较低,并行化方式构建可缩短运行时间,提高运行效率。在 Boston 和 Diabetes 样本量较少的数据集中,使用 2 线程并行运行时间效率较优,分别为 0.281 1 s、0.184 8 s,其加速比为 1.42 和 1.44;在 PRSA 和 AirQuality 数据集中,使用 4 线程并行运行时间效率较优,分别为 0.648 2 s、0.199 6 s,其加速比为 2.87、3.08;在 BikeSharing 和 Wine 数据集中,使用 8 线程并行化的运行时间效率较,分别为 0.437 4 s、0.333 7 s,其加速为 4.16 和 2.91。实验结果表明,本文模型的输入层适用于并行化运行,可显著提升模型运行效率,其并行度受限于与数据特征数相关的输入层节点数目。

4 结 语

本文融合深度学习和集成学习思想,提出了一种深度梯度提升树回归预测模型,以梯度提升回归树集成算法作为基础学习器,以深度学习的逐层表征学习策略将基础学习器组成多层级联结构,进行多层深度学习。输入层进行随机子空间特征学习,以应对复杂学习任务。多隐层结构逐层融合前一层特征学习结果并与原始特征进行逐层表征学习,依据学习变化率自适应确定隐层数。为保留学习器的多样性,隐层采用了 GBRT 和 XGBoost 两种不同的梯度提升算法。输出层中使用学习法结合策略对样本进行最终预测。模型采用了并行化方式训练各学习器以提高模型运行效率。实验结果表明,提出的预测模型其预测能力优于现有的集成学方法。

深度学习在处理复杂学习任务上具有显著优势,但深度神经网络计算成本高、超参数多。本文将传统机器学习方法与深度学习思想相结合,提供了一种可拓展的研究方向。

参 考 文 献

[1] 何晓群,刘文卿. 应用回归分析[M]. 4 版. 北京:中国人民大学出版社,2015.

[2] Friedman J, Hastir T, Tibshirani R. Additive logistic regression: A statistical view of boosting[J]. Annals of Statistics, 2000, 28(2): 337 – 407.

[3] Vapnik V N. The nature of statistical learning theory[M]. New York: Springer-Verlag, 1995: 85 – 86.

[4] Bianco V, Manca O, Nardini S. Electricity consumption forecasting in Italy using linear regression models[J]. Energy, 2009, 34(9): 1413 – 1421.

[5] Dudek G. Pattern-based local linear regression models for short-term load forecasting[J]. Electric Power Systems Research, 2016, 130(1): 139 – 147.

[6] An S, Liu W, Venkatesh S. Fast cross-validation algorithms for least squares support vector machine and kernel ridge regression[J]. Pattern Recognition, 2007, 40(8): 2154 – 2162.

[7] Fan G F, Peng L L, Hong W C, et al. Electric load forecasting by the SVR model with differential empirical mode decomposition and auto regression[J]. Neurocomputing, 2016, 173(1): 958 – 970.

[8] Cigizoglu H K, Alp M. Generalized regression neural network in modelling river sediment yield[J]. Advances in Engineering Software, 2006, 37(2): 63 – 68.

[9] Heddam S, Lamda H, Filali S. Predicting effluent biochemical oxygen demand in a wastewater treatment plant using generalized regression neural network based approach: A comparative study[J]. Environmental Processes, 2016, 3(1): 153 – 165.

[10] Shi J Q, Smith M R, Titterington D M. Bayesian regression and classification using mixtures of Gaussian processes[J]. International Journal of Adaptive Control & Signal Processing, 2003, 17(2): 149 – 161.

[11] Cakra Y E, Trisedya B D. Stock price prediction using linear regression based on sentiment analysis[C]//2015 International Conference on Advanced Computer Science and Information Systems, IEEE, 2015.

[12] Zhou Z H. Ensemble methods-foundations and algorithms [M]. Oxfordshire: Taylor & Francis, 2012.

[13] Baszczynski J, Stefanowski J. Neighbourhood sampling in bagging for imbalanced data[J]. Neurocomputing, 2015, 150(2): 529 – 542.

[14] Zięba M, Tomczak S K, Tomczak J M. Ensemble boosted trees with synthetic features generation in application to bankruptcy prediction[J]. Expert Systems with Applications, 2016, 58(10): 93 – 101.

[15] Tang B Z, Chen Q C, Wang X, et al. Reranking for stacking ensemble learning[C]//2010 International Conference on Neural Information Processing: Theory and Algorithms, Springer-Verlag, 2010: 575 – 584.

[16] Breiman L. Random forests[J]. Machine Learning, 2001, 45: 5 – 32.

[17] Wang Y Z, Feng D W, Li D S, et al. A mobile recommendation system based on logistic regression and gradient boosting decision trees[C]//2016 International Joint Conference on Neural Networks (IJCNN), IEEE, 2016: 1896 – 1902.

[18] Chen T Q, Guestrin C. XGBoost: A scalable tree boosting system[C]//Proceedings of 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM Press, 2016: 785 – 794.

[19] LeCun Y, Bengio Y, Hinton G. Deep learning[J]. Nature, 2015, 521: 436 – 444.

[20] Montufar G F, Pascanu R, Cho K, et al. On the number of linear regions of deep neural networks[C]//Annual Conference on Neural Information Processing Systems. MIT Press, 2014: 2924 – 2932.

[21] Polson N G, Sokolov V O. Deep learning for short-term traffic flow prediction[J]. Transportation Research (Part C): Emerging Technologies, 2017, 79: 1 – 17.

[22] Hu Q, Zhang R, Zhou Y. Transfer learning for short-term wind speed prediction with deep neural networks[J]. Renewable Energy, 2016, 85: 83 – 95.

[23] Qi Z Q, Wang B, Tian Y, et al. When ensemble learning meets deep learning: A new deep support vector machine for classification[J]. Knowledge-Based Systems, 2016, 107(C): 54 – 60.

[24] Zhou Z H, Feng J. Deep Forest: Towards an alternative to deep neural networks[EB]. [2018 – 06 – 02]. arXiv: 1702.08835v2, 2017.

[25] Schapire R E. The strength of weak learnability[C]//Proceedings of the Second Annual Workshop on Computational Learning Theory, 1989: 383.

[26] Friedman J H. Greedy function approximation: A gradient boosting machine[J]. Annals of Statistics, 2001, 29(5): 1189 – 1232.

[27] Bernard S, Heutte L, Adam S. On the selection of decision trees in random forests[C]//Proceedings of the 2009 International Joint Conference on Neural Networks, 2009: 790 – 795.

(上接第 98 页)

[10] Bahdanau D, Cho K, Bengio Y. Neural machine translation by jointly learning to align and translate[EB]. arXiv: 1409.0473, 2014.

[11] Chen H M, Sun M S, Tu C C, et al. Neural sentiment classification with user and product attention[C]//Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. ACL, 2016: 1650 – 1659.

[12] Zhou P, Shi W, Tian J, et al. Attention-based bidirectional long short-term memory networks for relation classification [C]//Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics. 2016, 2: 207 – 212.

[13] Zhang X, Lecun Y. Which encoding is the best for text classification in Chinese, English, Japanese and Korean? [EB]. arXiv: 1708.02657, 2017.

[14] Silva R M, Almeida T A, Yamakami A. MDLText: An efficient and lightweight text classifier[J]. Knowledge-Based Systems, 2017, 118(2): 152 – 164.

[15] Zhang X, Zhao J, LeCun Y. Character-level convolutional networks for text classification[C]//Advances in neural information processing systems, 2015: 649 – 657.

[16] Zhou P, Qi Z Y, Zheng S C, et al. Text classification improved by integrating bidirectional LSTM with two-dimensional max pooling[EB]. arXiv: 1611.06639, 2016.

[17] 李洋,董红斌. 基于 CNN 和 BiLSTM 网络特征融合的文本情感分析[J]. 计算机应用, 2018, 38(11): 29 – 34.

[18] Wen Y, Zhang W N, Luo R, et al. Learning text representation using recurrent convolutional neural network with highway layers[EB]. arXiv: 1606.06905, 2016.

[19] 李锋刚,梁钰,高晓智,等. 基于 LDA-wSVM 模型的文本分类研究[J]. 计算机应用研究, 2015, 32(1): 21 – 25.