《机器学习导论》期末大作业-ETA 作业报告

ETAshiliuli2020
施毅，刘海天，李瑞翔
(181220048, 181220034, 181220029)
synju@outlook.com, 181220034@smail.nju.edu.cn, 181220029@smail.nju.edu.cn

摘 要：本实验采用两层集成的方法，它是单个集成学习模型与 stacking 模型融合技术的结合，可以看作 Bagging 学习方法和 Boosting 类学习方法的推广和融合。该方法被用于预测船运到达时间的模型的训练。我们小组的成员们使用五折交叉验证、模型间横向对比、以及比赛反馈的得分，对所提出的算法进行了测试。结果表明，相对于单一的集成学习模型，使用该方法得到的均方误差更低，性能得到一定的改善。
关键词：集成学习；stacking；船运到达时间预测

1 介绍

船运到达时间预测是最近几年来人工智能算法应用中刚刚兴起的问题。在企业全球化业务体系中，海运物流凭借其成本低的巨大优势，成为其最重要的一项支撑，而伴随着出海电商的热潮，海运物流又将在其中占据重要地位。

船运到达时间作为十分重要的一项数据，出于海运物流速度慢的自身局限，以及恶劣天气、港口拥挤、空间或设备短缺、船舶航线变化，还有错误录入等人为导致的问题，船运公司所能给出的该数据的预测值存在缺失和偏差较大的问题。而这个问题预期可以在大数据和人工智能技术快速发展的新时代背景下得到一定缓解。运用大数据分析和人工智能算法技术，数据供应公司可以同船运公司合作，提供更精准的船运到达时间的预测，从而提高海运物流行业的可靠性和供应链的整体性能，为供应链提供精准部署的数据基础。这将将能够为出海企业建立更精准的全球化供应链方案提供关键的技术支持。

对于船运到达时间预测这类问题，具体开展的研究十分有限。而基于该类问题提供的数据：地理空间数据，对于地理空间数据的回归预测，已经有了大量的研究，当中已经探索出了多类技术，例如隐马尔科夫模型[4]、卡尔曼滤波器[2]、支持向量回归[6]、k 近邻[3]模型和一些数据驱动模型如人工神经网络(ANNs)[5] [1]。

但是，上述方法均存在不足，特别是在学习船运公司给出的存在大量缺失值以及异常值的数据时，上述方法所利用到的模型在鲁棒性上很难得到保证。同时，由于所提供的数据之间原本就存在极高的相似度，一些单个学习能力较强的模型在船运到达时间预测问题中存在很高的过拟合的风险，得不到较好的泛化性能。

集成学习作为当下最有前途和最流行的机器学习方法之一，在交通运输领域的问题中的表现是相当不错的。在该领域中，例如 Random Forest、Gradient Boosting machine 和 BoostedRegression tree 的基于多种简单决策树的集成的树集成模型[1]已经获得了成功的应用。由于船运到达时间预测问题与该领域所探讨的一些预测问题存在相似之处，故集成学习在处理海运物流数据方面的能力很值得期待和观察。

我们针对本次的竞赛题目，在对所提供数据进行细致的预处理以及特征提取的基础上，提出了“两层集成”的想法。具体而言，我们利用 stacking 策略，将原本就是集成学习模型的学习器 XGBoost，GradientBoost 以及随机森林作为基学习器，在此基础上利用线性回归模型作为元模型(meta model)实现第二层集成，找到了解决单一学习器鲁棒性差和拟合能力过强的问题的有效途径。我们用赛题所提供的测试数据对新方法进行了测试，结果表明利用 stacking 策略进行融合之后的集成模型，其泛化性能较于各个组成部分的模型均有较高提升。

本报告的结构安排如下：第二部分将介绍我们小组对赛题给出的数据进行建模，具体来说是特征工程处理的过程和一些手段；第三部分将探讨我们在实验中所运用到的基本理论与方法，特别是我们在为提高模型泛化能力方面所使用的模型融合的方法；第四部分中会逐个介绍在模型训练中涉及到

的各个模型的算法与各自的优缺点；第五部分将展示实验的结果，对于各类单一学习器的性能给出评价，给出我们在模型的参数选取中进行的尝试，并对单个学习器和采用模型融合后的学习器的性能进行横向比较；第六部分会对我们的实验结果进行简要的评价和说明，并分析实验中采取的方法的优势与不足；第七部分将整体概括我们小组本次实验做出的工作，总结实验成果，并给出了我们现有工作中的不足之处和今后可以改进的方向。

2 模型

原始数据为每个运单在船运的过程中，所在船产生的 GPS 位置的相关信息。一个运单由多行数据组成，很难直接对其建模。因此，需要通过特征工程进行处理。

本实验特征工程主要分为两大步。首先对训练数据进行清洗，主要包括对速度、方向等缺失值和异常值的处理。其次，特征提取为特征工程的主要部分，需要根据同一份订单的所有 GPS 历史数据创造出一系列新的基于订单的“泛化”特征。具体而言，数据可以被原始特征’loadingOrder’(订单编号)划分为若干独立的数据块。每个数据块都离散地记录了航行的信息，其中我们认为速度、方向、航行状态以及坐标相对比较重要。我们将同一订单下的各个历史 gps 数据进行特征整合，使得在训练集中每一个航运订单只对应一条样本。处理后的特征空间为 R_{14},也就是说训练集中的每个样本包含有 14 个属性，如表 1 所示。

特征	说明
longitude	该份订单的 gps 信息中，最初经度坐标
latitude	该份订单的 gps 信息中，最初纬度坐标
lastLongitude	该份订单的 gps 信息中，最末经度坐标
lastLatitude	该份订单的 gps 信息中，最末纬度坐标
anchorTime	我们认为船舶在海运过程中的停滞时间对于整个航程的时间有较大影响，故选取停滞时间作为一个特征
manDis	航程的总距离
maxSpeed, minSpeed, meanSpeed medianSpeed	新创造出的特征，用于表征同一份订单的更全面的速度信息
maxDirdiff, minDirdiff, meanDirdiff, medianDirdiff	新创造出的特征，用于表征该份订单中，船舶在行驶过程中的方向同整体方向(出发点和到达点的位置关系)的差异程度

表 1 提取特征及说明

训练集标签选择时间间隔（timeInterval），其分布如图 1 所示，基本符合正态分布。

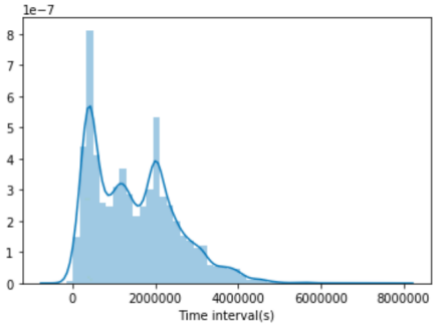

图 1 训练集标签分布

此外，我们舍弃了手工录入数据和缺失值占极大多数的历史运单数据，首先是因为该部分数据可能包含大量噪声；其次是因为若利用历史运单数据构造特征，可能会导致我们特征提取内部各个订单之间存在不一致性，而这是因为我们希望统一采取每个订单的最大时刻对应的经纬度作为起点和终点，而不是选择目标港口为终点，故引入历史运单数据的话将造成不同订单样本特征提取方式的不一致。

这样将特征和标签都转换成连续的实数，可以将其建模成一个简单的回归问题：

设数据集为 $D = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{m}$，其中 $\boldsymbol{x}_i \in R^{14}$，$y_i \in R$，目标是训练出一个模型 $h: \boldsymbol{x} \rightarrow y$，使其尽可能准确的预测实值标记。

3 方法与方法分析

3.1 方法

在对船运到达时间的预测过程中，由于所给的训练数据很多，为了提高预测的精度，又不希望牺牲模型的泛化能力，我们采取了模型融合的方法。

具体地，我们采取了 Stacking 的策略进行模型融合。Stacking 策略的思想是一种有层次的融合模型，以两层为例，第一层由多个基学习器组成，其输入为原始训练集，第二层的模型则是以第一层基学习器的输出作为训练集进行再训练，从而得到完整的 Stacking 模型。

而在本次训练任务中，我们首先比较了广泛适用于回归任务的多种基学习器，并选择了表现较好的 XGBoost, RandomForest，Bagging 以及 LightGBM 四种集成学习模型作为基学习器。同时，我们选取了线性模型 LinearRegression 作为元学习器，通过学习训练，给基学习器的预测结果上赋予权重，使最后的预测最为准确。具体模型如图 2 所示。

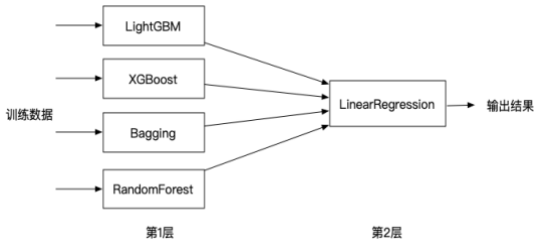

图 2 本实验 Stacking 模型图

3.2 方法分析

Stacking 先从初始数据集中训练出初级学习器，然后“生成”一个新数据集用于训练次级学习器。在这个新数据集中，初级学习器的输出被当做样例输入特征，而初始样本的标记仍被当做样例标记。

采取 stacking 策略对进行模型融合，我们的方法的优势主要有以下几点:

1) 以集成学习模型作为基学习器，通过第二层的元模型又进行了一次集成，相当于进行了两层的集成，比单一的学习器或学习模型都具有更好的泛化性能；

2) 选择 stacking 策略进行模型融合，在较大的数据集上具有更高的鲁棒性。

4 算法与算法分析

4.1 算法

如上文所说，在初级学习器中，我们采取了 XGBoost，lightGBM，Bagging 和 RandomForest 的方法.

其中 Xgboost 算法，lightGBM 算法都属于集成学习中的 boosting 类算法中的 gbdt（GradientBoostingDecisionTree）的变种。boosting 类算法的工作机制类似，先从初始训练集中训练出一个基学习器，再根据基学习器的表现对训练样本分布进行调整，使得先前基学习器做错的训练样本在后续受到更多关注，然后基于调整后的样本分布来训练下一个基学习器;如此重复进行。

此外，我们还选择了 bagging 和随机森林作为 stacking 的初级学习器。这两种模型都采用随机采样的方法保证了基学习具有比较大的差异。

下面我们简要地介绍一下用到的算法。

1) Gbdt

XGBoost 和 lightGBM 都是 Gbdt (GradientBoostingDecisionTree)的变种，因此我们先来介绍一下 Gbdt。Gbdt 中，dt 是指 Decision Tree 表示使用决策树作为基学习器，gb 表示梯度提升，因为在传统的 gbdt 中在第 i 轮的迭代中，使用前$i-1$的梯度作为当前残差进行拟合。

第 m 轮训练之后，gbdt 的结果如下:

$$f_m(x) = f_{m-1}(x) + \beta_m b(x;\gamma_m)$$

Gbdt 的格式化损失函数如下:

$$L\big(y, f_m(x)\big) = L\big(y, f_{m-1}(x) + \beta_m b(x;\gamma_m)\big)$$

对损失函数进行一阶展开后，可以得出第 m 棵树使用前 m-1 棵的负梯度作为残差，每次只需对负梯度进行拟合。

2) XGBoost

xbgoost 作为 gbdt 的改进，在损失函数中加入了正则项，用以控制模型的复杂度，xgboost 算法的目标函数为：

$$obj^{(t)} = \sum_{i=1}^{n} l\left(y_i, \hat{y}_i^t + \sum_{i=1}^{t} \Omega\left(f_i\right)\right)$$

$$= \sum_{i=1}^{n} l\left(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)\right) + \Omega\left(\mathrm{f_i}\right) + constant$$

对损失函数进行二阶泰勒展开，求最小化参数，可以得到最小损失：

$$\hat{l}^{(t)}(q) = -\frac{1}{2}\sum_{j=1}^{T}\frac{\left(\sum_{i\in \mathrm{I_j}} g_i\right)^2}{\sum_{i\in I_j} h_i + \lambda} + \gamma T$$

Xgboost 在过拟合方面采取了缩减(Shrinkage)的措施，即每棵树的结果乘以一个较小的常数 eta。为了提高算法的效率和基学习器的差异，并防止过拟合，xgboost 还借鉴了随机森林的做法，支持列抽样，即在生成树的过程中，随机选取包含 K 个特征的子集，再从这个子集中选择一个属性进行划分。

3) LightGBM

lightGBM 核心思想与 Xgboost 相同，都是采用损失函数的负梯度作为当前决策树的残差近似值，去拟合新的决策树。

但它相较于 Xgboost 的不同主要有两点，在树的切分策略上，XGBoost 是采用 level-wise 的方法，而 LightGBM 则是采用带深度限制的按叶子生长策略. 相较于 level-wise，leaf-wise 在叶子数量一样时，能降低误差，得到更好地精度，加上限制防止生成深树而在小数据集上造成过拟合。

另外，lightGBM 使用直方图方法将连续的特征值离散化成一个个 bin，提升了效率，节省了空间。

4) Bagging

Bagging 算法利用 bootstrap sampling 的采样方式从训练集中选取 m 个样本，由于是可放回采样，

初始训练集中有的样本在采样集里多次出现，有的则从未出现。

照这样采样出 T 个含 m 个样本的训练集，基于每个采样集训练出一个基学习器，再将这些基学习器进行结合，就是 bagging 的基本流程。

5)Random Forest

随机森林算法作为 Bagging 的一个扩展变体，进一步在决策树的训练过程中引入了随机属性选择，对基决策树的每个结点，先从该结点的属性集合中随机选择一个包含 k 个属性的子集，再从这个子集中选择一个最优属性进行划分。

4.2 算法分析

Bagging 作为一个很高效的集成学习算法，它的复杂度基本与基学习器的复杂度同阶，且能不经修改地用于多分类、回归任务。且随机采样让其拥有约 36.8%的样本可用作验证集来对泛化性能进行“包外估计”，有利于增强学习器的泛化性能。

随机森林算法作为 Bagging 的一个扩展变体，进一步在决策树的训练过程中引入了随机属性选择，大大增强了基学习器的多样性，随着学习器数目的增加，随机森林通常可以收敛到较低的泛化误差。由于随机森林使用了使用了行采样和列采样技术，使得每棵树不容易过拟合；并且是基于树的集成算法，由于每棵树的采用的数据差别较大，在进行 embedding 的时候可以更好的降低模型的方差，故随机森林具有较好的鲁棒性。

前两种集成算法主要是增大分歧来保证集成的性能，后两种 boosting 算法则是通过不断拟合残差减小误差来保证集成的性能。

Xgboost 支持工具并行，在特征的粒度上，可以同时计算各个特征的增益；正则项以及缩减和列抽样方法的加入可以有效防止其过拟合，增大其泛化能力，且能够加速计算。

Lgbm 同样支持工具并行相较于 Xgboost，占用的空间更少且更加快速，且深度受限的 leaf-wise 特征分裂方法可以防止树过深的问题。

这几种集成学习方式都能在消耗资源较少的情况下利用较为简单的基学习器获得较强的泛化能力。

最后，根据 stacking 策略对以上四种算法的结果进行再学习，对以上四种算法得到的模型进行模型融合，进一步发挥了集成学习的优势。

5 数值结果

5.1 仿真环境

本文所做的实验选用华为云在线开发环境 ModelArts，规格为 CPU 8 核 32GiB，开发环境为 jupyter notebook，软件平台为 python3.6（kernel 为 XGBoost-Sklearn）。

实验训练数据主要选择大赛提供的 train0523（约 1.5 亿行），并辅之以 port 和 loadingOrderEvent 数据。训练集经特征工程处理后共有 18951 个实例，14 个特征。

5.2 实验评估

评估指标选择均方误差（MSE）。记真实值的时间戳为 ATA，预测值的时间戳为 TA，单位：秒，计算如下

$$\mathrm{MSE}=\frac{\sum(\mathrm{ETA}/3600-\mathrm{ATA}/3600)^2}{\mathrm{ETA\_NUM}}$$

实验采用 5 折交叉验证法，结果取在验证集上的 MSE 的均值。

5.3 不同模型的比较

选择常用回归模型，并重点比较了各线性模型与集成学习。在默认参数下，其结果如表 2 和图 3 所示。

	模型	MSE
决策树	DecisionTreeRegressor	3955.5847
支持向量机	SVR	29079.2793
K 近邻	KNeighborsRegressor	6167.6550
线性模型	LinearRegression	10212.7781
	Lasso	10217.6705
	BayesianRidge	10213.4750
	LassoLarsIC	10213.0865
集成学习	XGBoost	4951.7477
	GradientBoostingRegressor	4915.3874
	LightGBM	3412.3204
	BaggingRegressor	2639.7264,
	RandomForestRegressor	2767.6931

表 2 不同模型的 MSE 结果

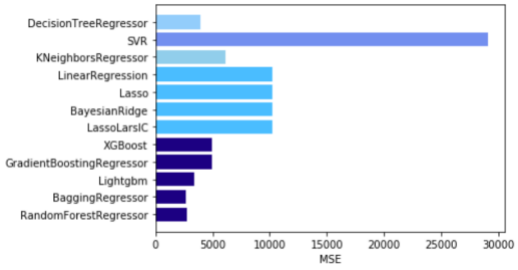

图 3 不同模型比较结果

结果表明，在默认参数的条件下，集成学习的结果明显优于其他方法，其中随机森林和 bagging 表现最佳。这表明集成学习的优势，即通过将多个学习器结合，能够得到更加优越的泛化性能，比单

一学习器的表现性能更加。

而其中各个线性模型都表现类似，MSE 均为 1w 左右，表明本问题的特征并不是完全符合线性的特征。尽管部分特征如航线距离、滞留时间等从直观上来说与时间间隔成线性关系，但其他的一些特征如经纬度等，可能引起了一定的偏差。

5.4 模型参数选择

根据上述比较结果，选择性能较好的几个模型：LightGBM、XGBoost、Bagging 和 RandomForest。本文没有参数选择上做过多的实验，大多采用模型的默认值，或着参考往届 kaggle 比赛优秀[7]的模型。发现通过增大基学习器的数目，XGBoost 的性能得到很大的提高，从原来的 4951 减至 2994。其余几个调参后没有显著的提高。

5.5 模型融合结果

将上述四个模型作为初级学习器，并使用 LinearRegression 作为次级学习器，进行 stacking 模型融合，并与平均法（averaging）进行比较。比较结果如图 4 所示。

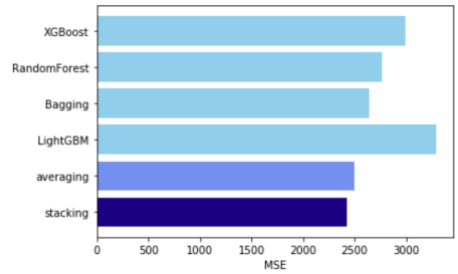

图 4 模型融合结果与原学习器结果比较

结果表明，通过第二层的集成学习，最终的表现性能确实要优于单一的集成模型。虽然从理论上来说，基学习器选择“弱学习器”足以表现较好的性能，但考虑到本实验中数据量较大，而且可能会存在较大的噪声，所以选择少量的强学习器进行训练。

相比 Averaging，Stacking 的结果有少量的提升，但并不明显。Stacking 的优势在于，它使用交叉验证，一定程度上避免了过拟合现象，并给予不同模型以不同的权重，能够更好地拟合目标。

6 讨论

我们没有穷尽搜索所有模型，而是在前人工作的基础上，选择经典的模型进行比较。选择若干较好的模型，通过调参得到最优的模型，并将其作为初级学习器，使用 stacking 策略进行模型融合。我们的方法的优势主要有两点：(1)以集成学习模型为基学习器，再此之上再进行了一次集成，相当于进行了两层的集成，比单一的学习器或单一集成学习模型都具有更好的泛化性能；(2)选择 Stacking 策略，而不是 Averaging 策略，其中使用交叉验证，一定程度上避免了过拟合现象，并给予不同模型以不同的权重，在较大的数据集上具有更高的鲁棒性。

7 结论

我们最终的模型包含两层集成：将 4 个集成学习模型（LightGBM、XGBoost、Bagging 和 RandomForest）作为初级学习器，并使用 stacking 策略进行模型融合。通过对 1.9 万条运单的训练，我们的均方误差降为 2450 左右。性能高于单一的集成学习器，且具有更好的鲁棒性。

使用该模型在测试集上进行测试，提交结果却达 4265.8032，与训练集的结果相差近两倍，还有较大的改进空间。我们认为，主要原因还是在于训练数据的特征选取上。由于计算资源和时间有限，在处理每份订单的起点和终点时，只是简单地将订单的最早时刻和最大时刻对应的经纬度作为起点和终点，而不是选择目标港口为终点。事实上，船快驶入港口时，速度可能会减慢，与选取的特征可能会产生一定的偏差。另外，由于时间关系，我们没有尝试神经网络的方法，今后可以尝试将神经网络与集成学习相结合，或许会有更好的效果。

致谢 在此，我们向一学期认真教学的老师们，以及认真批改作业、耐心给予帮助的助教们表示感谢。

参考文献

[1] Oleh Bodunov, Florian Schmidt, André Martin, Andrey Brito, and Christof Fetzer. 2018. DEBS '18: Proceedings of the 12th ACM International Conference on Distributed and Event-based SystemsJune 2018 Pages 198–201

[2] Xiang Fei, Chung-Cheng Lu, and Ke Liu. 2011. A bayesiandynamic linear model approach for real-time short-term free-way travel time prediction.Transportation Research Part C:Emerging Technologies19, 6 (2011), 1306 – 1318.

[3] Jiwon Myung, Dong-Kyu Kim, Seung-Young Kho, and Chang-Ho Park. 2011. Travel Time Prediction Using k NearestNeighbor Method with Combined Data from Vehicle DetectorSystem and Automatic Toll Collection System.Transporta-tion Research Record: Journal of the Transportation ResearchBoard2256 (2011), 51–59.

[4] Christopher Tay, Kamel Mekhnacha, and Christian Laugier. 2012.Probabilistic Vehicle Motion Modeling and Risk Estimation.Springer London, London, 1479–1516.

[5] J.W.C. van Lint, S.P. Hoogendoorn, and H.J. van Zuylen. 2005.Accurate freeway travel time prediction with state-space neuralnetworks under missing data.Transportation Research Part C:Emerging Technologies13, 5 (2005), 347 – 369

[6] Chun-Hsin Wu, Chia-Chen Wei, Da-Chun Su, Ming-Hua Chang,and Jan-Ming Ho. 2003. Travel time prediction with supportvector regression. InProceedings of the 2003 IEEE InternationalConference on Intelligent Transportation Systems,

Vol. 2. 1438–1442 vol.2. [10] Chun-Hsin Wu, Chia-Chen Wei, Da-Chun Su, Ming-Hua Chang,and Jan-Ming Ho. 2003. Travel time prediction with supportvector regression. InProceedings of the 2003 IEEE InternationalConference on Intelligent Transportation Systems, Vol. 2. 1438–1442 vol.2. https://doi.org/10.1109/ITSC.2003.1252721
[7] https://www.kaggle.com/serigne/stacked-regressions-top-4-on-leaderboard
[8] https://www.kaggle.com/dansbecker/xgboost
[9] 周志华，机器学习，清华大学出版社，2016

附录

A. 工程文件

项目工程文件主要由 4 个 ipynb 文件组成

- preprocess_part1.ipynb：对原始数据分批（每 200w 行）进行预处理和特征提取，并保存至 74 个 csv 文件中
- preprocess_part2.ipynb：对 74 个文件进行合并，将同一份订单的数据合并，并进行二次预处理。
- training.ipynb：进行模型的训练和比较
- result.ipynb：通过加载训练好的模型进行预测

B. 关键代码说明

- preprocess：通过对训练集进行多线程处理和分块处理，提高处理速度
- add_feature：合并相同订单，并添加特征
- 类 Averaging：平均法模型融合
- 类 Stacking：Stacking 法模型融合

C. 小组队员及分工

姓名	学号	分工
施毅	181220048	特征工程（预处理）、模型训练、模型融合
李瑞翔	181220029	特征工程(特征提取)、模型训练、结果生成
刘海天	181220034	特征工程（预处理）、模型训练、比较分析