Report

12110644 周思呈

目录

1 问题描述

给定一组来自1994 Census bureau database by Ronny Kohavi and Barry Becker (Data Mining and Visualization, Silicon Graphics)的数据，要求训练一个模型，能够预测新产生样本的收入是否超过每年$50K。数据包括特征和标签两个部分，特征包括年龄、工作种类、观测代表数量、教育程度、教育时长、婚姻状况、职业、家庭关系、种族、性别、固定资产收益和损失、每周工作时长、国籍。

2 数据预处理

2.1 非数值属性编码

2.1.1 有序特征赋值

education这一特征可以按照教育程度高低进行编码。查资料得，美国教育程度属性值可如下排列：

1. Doctorate（博士学位）
2. Masters（硕士学位）
3. Professional School（专业学校，如法学院或医学院）
4. Bachelors（学士学位）
5. Associate's Degree in Academic Studies（学院学位）
6. Associate's Degree in Vocational Studies（职业学位）

7. Some College（部分大学教育，但没有获得学位）
8. High School Graduate（高中毕业）
9. 12th Grade（12年级）
10. 11th Grade（11年级）
11. 10th Grade（10年级）
12. 9th Grade（9年级）
13. 7th-8th Grade（7年级至8年级）
14. 5th-6th Grade（5年级至6年级）
15. 1st-4th Grade（1年级至4年级）
16. Preschool（学前班）

按照上述顺序对该属性进行赋值替换，教育程度越高所对应的值越大。

2.1.2 类别特征独热编码

使用独热码将每一特征值扩展为一个特征，其中每个样本对应属性值为0或1，对应表示该属性值是否为真。这种编码方式能保证不同类别之间没有共线性，在后续计算欧氏距离等过程中使不同类别之间距离相同，更加合理。

2.2 归一化

独热编码后的数据中只有0和1两个值，但age，education，education.num，capital.gain，capital.loss，hours.per.week这些属性值却大小不等。为了使各个属性在最开始训练时具有公平的贡献比，使用Z-Score归一化将数据转换为均值为0、标准差为1的分布。该方法公式如下：

$$X\_scaled = (X - X.mean)/X.std$$

2.3 特征筛选

通过计算特征之间的皮尔逊相关系数，将相关系数较高的特征合并，从而达到一定的降维效果。

皮尔逊积矩相关系数（Pearson product-moment correlation coefficient）是两个变量的协方差与其标准差的乘积之比，从几何意义上说表示两个变量线性相关的程度。具体计算公式如下：

$$\rho_{X,Y} = \frac{E(XY) - E(X)E(Y)}{\sqrt{E\left(X^2\right) - (E(X))^2}\sqrt{E\left(Y^2\right) - (E(Y))^2}}$$

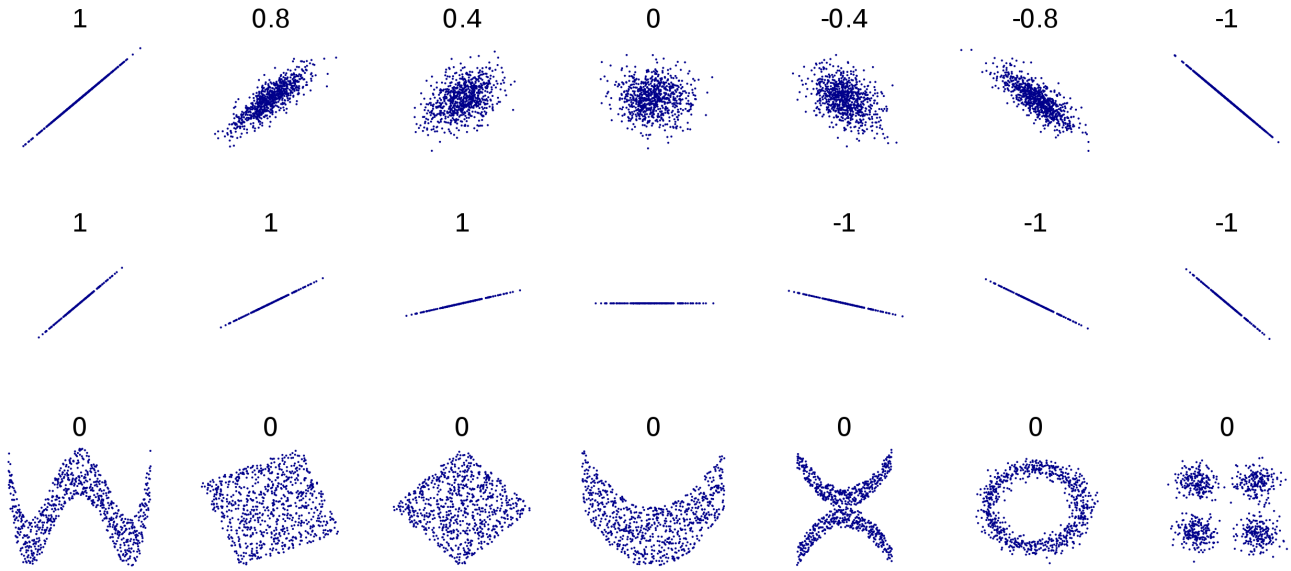


Fig2.1，不同相关系数下变量的分布情况。

对数据集中特征计算相关系数，筛选出绝对值大于0.6的如下结果：

特征1	特征2	相关系数
workclass_?	occupation_?	0.9979335562938918
marital.status_Married-civ-spouse	marital.status_Never-married	-0.6444622054820798
marital.status_Married-civ-spouse	relationship_Husband	0.8932578449525168
race_Black	race_White	-0.7859528210492007
sex_Female	sex_Male	-1.0
education	education.num	0.9951651595652403

将上述相关性较强的特征合并之后获得新的特征矩阵。

2.4 缺失值处理

经过初步分析，发现训练集中workclass和occupation这两个feature（重叠）存在着5.6%左右的数据缺失，而native.country存在着1.9%左右的数据缺失。进一步分析发现，testdata数据集中同样存在着5.6%左右的样本缺失workclass和occupation，1.6%左右的样本缺失native.country。在实际情况中，确实可能出现不明确工作种类的情况，因此将这部分缺失值保留作为单独一类，不作进一步处理。而国籍信息中，90%的样本值都为United-States，剩下的各个国家占比均较小，与缺失值情况类似，因此也不做进一步处理。

在实际训练过程中尝试了赋值为众数、直接删除和不作处理三种方式，发现对结果并无特别大影响，因此最后选择不作处理。

3 训练模型

3.1 决策树

这一部分的代码参见Decision_Tree.ipynb文件。

3.1.1 方法

决策树是一种简洁有效的分类方法，通过每次贪心选择最大信息增益的属性完成分类。ID3算法通过Entrophy计算信息增益：

$$\operatorname{Entropy}(S)=\sum_{i=1}^{c}-p_i\log_2 p_i$$

$$\operatorname{Gain}(S,A)=\operatorname{Entropy}(S)-\sum_{v\in\operatorname{Value(A)}}\frac{|S_v|}{|S|}\operatorname{Entropy}\left(S_v\right)$$

CART算法通过Gini函数计算信息增益：

$$\operatorname{Gini}\ =1-p_{\oplus}^2-p_{\ominus}^2$$

采用后剪枝的方法防止过拟合：利用测试集计算去掉某一子树后模型的预测准确性，选择准确性最高的剪枝方法。

3.1.2 训练过程及结果

首先用sklearn.tree中DecisionTreeClassifier的默认参数训练一个模型作为baseline。默认参数的信息增益函数使用Gini。采用五折交叉验证获取该模型平均准确性。最终Average score: 0.8128727720922149。

DecisionTreeClassifier提供了三种split criterion函数，分别为gini，entropy和log_loss。它还提供了splitter参数用于在每个节点上选择分割的策略,支持的策略是选择最佳分裂的“最佳”策略和选择最佳随机分裂的“随机”策略。采用网格搜索和五折交叉验证的方法对上述策略的组合进行训练和检验，获得如下结果：

params	mean_test_score
{'criterion': 'gini', 'splitter': 'best'}	0.813909
{'criterion': 'gini', 'splitter': 'random'}	0.802556

params	mean_test_score
{'criterion': 'entropy', 'splitter': 'best'}	0.810234
{'criterion': 'entropy', 'splitter': 'random'}	0.807492
{'criterion': 'log_loss', 'splitter': 'best'}	0.810234
{'criterion': 'log_loss', 'splitter': 'random'}	0.807492

Best parameters: {'criterion': 'gini', 'splitter': 'best'}
Best score: 0.813908786082114

测试得到最优的参数组合是criterion=gini，splitter=best。

然后对获得的决策树进行剪枝。使用 `cost_complexity_pruning_path` 方法计算出一条路径，该路径上的每个节点对应一个不同的 `ccp_alpha` 值，用于控制剪枝的参数。遍历每个 `ccp_alpha` 值，创建新的决策树分类器 `clf`，并设置 `ccp_alpha=ccp_alpha`。使用训练集数据 `X_train` 和目标变量 `y_train` 对新的决策树模型进行训练，获得新模型的准确率。从中选取准确率最高的模型。最后得到的参数结果是Best alpha = 0.0001755087974549532，Best accuracy = 0.8613731081377495。可以看到，剪枝后的模型避免了过拟合，因此在测试集上获取了良好的效果。

训练出的决策树可以输出每个特征对应的重要程度，其结果如下（可以看出高科技产业还是很重要的）：

特征	重要程度
occupation_Tech-support	0.32124405701719755
native.country_Vietnam	0.20671050720874562
native.country_Yugoslavia	0.20434391157558723
relationship_Own-child	0.06752021293414374
native.country_Trinadad&Tobago	0.051128750110392245
age	0.05237420719939163
education	0.036639249148847086
native.country_United-States	0.007873806420248163
workclass_Self-emp-inc	0.009586453712542305
occupation_Armed-Forces	0.013349805374056738
occupation_Handlers-cleaners	0.0041636784048882805
occupation_Sales	0.003979801111144255
occupation_Other-service	0.00397938835699604
occupation_Craft-repair	0.003435368620353291
native.country_South	0.0013728372206275718
occupation_Exec-managerial	0.0015585627112692517
occupation_Prof-specialty	0.0015142323157715492
workclass_Private	0.0021091075553618093
workclass_?:	0.002292878195014191
occupation_Farming-fishing	0.0011035498779476671
occupation_Protective-serv	0.0007465861473625799
native.country_United-States	0.007873806420248163

3.2 集成学习

这一部分的代码参见Ensemble.ipynb文件。

3.2.1 方法

集成学习通过多数投票将许多单个分类器的预测组合在一起形成新的预测结果。其中每个单个分类器只需要有高于0.5的预测准确率。简单分类器的多数投票能有效减小模型variance，达到bias和variance的相对平衡。

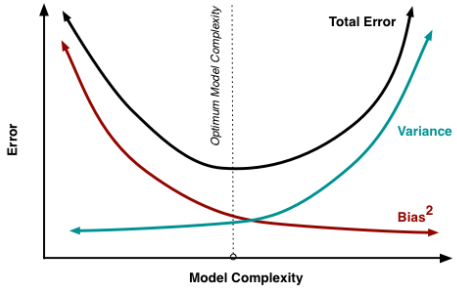


Fig3.1，努力达到vias和variance的平衡。

集成学习的方法主要有Bagging和Boosting。Bagging中预测函数是均匀平等的，但在Boosting中预测函数是加权的，在建立预测模型时更加灵活。此处采用XGBoost方法进行分类预测。

XGBoost在GBDT(Gradient Boosting Decision Tree)的基础上进行算法优化以提升计算效率。GBDT中，每个弱分类器都去拟合之前分类器产生的误差函数对预测值的残差，最后所有弱分类器的结果相加等于预测值。XGBoost在GBDT的基础上优化了目标函数的定义[1]:

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

用泰勒展开近似后:

$$g_i=\partial_{\hat{y}^{(t-1)}}l\left(y_i,\hat{y}^{(t-1)}\right),h_i=\partial_{\hat{y}^{(t-1)}}^2l\left(y_i,\hat{y}^{(t-1)}\right)$$

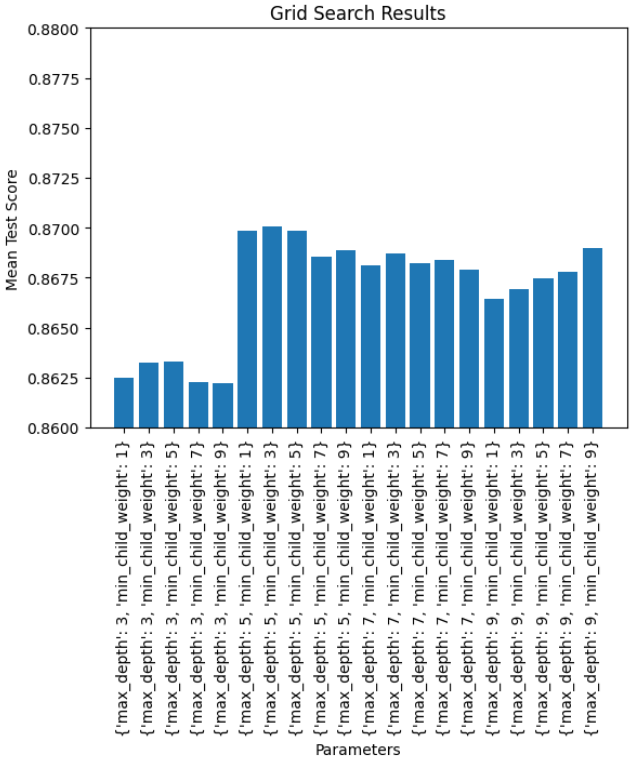
$$Obj^{(t)}\simeq\sum_{i=1}^{n}\left[l\left(y_i,\hat{y}_i^{(t-1)}\right)+g_if_t\left(x_i\right)+\frac{1}{2}h_if_t^2\left(x_i\right)\right]+\Omega\left(f_t\right)+constant$$

3.2.2 训练过程及结果

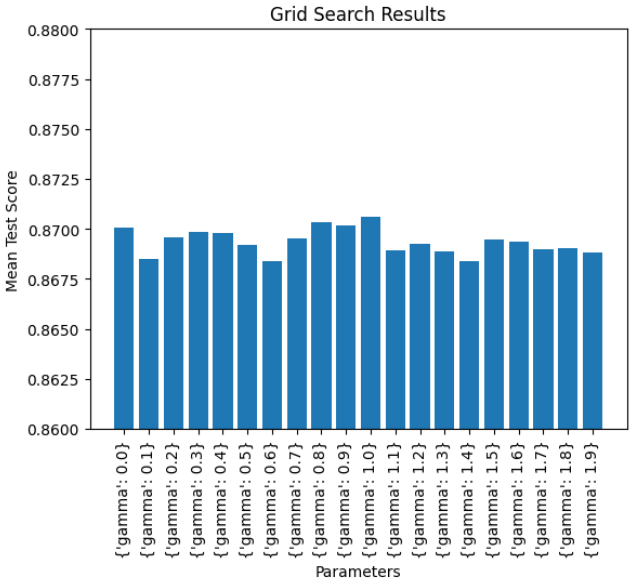
XGBoost相比单纯的决策树模型有更多参数可以调整。以下调参过程参考【XGBoost参数调优完全指南】[2]。

首先随意选取一组参数作为baseline。此处选择最大树深=3，学习率=0.3，迭代次数=100。获得Accuracy: 0.8725597718797982。在训练过程中可以看到，即便是随意选取了一组参数，xgb的分类效果也已经好于前面剪枝过后决策树的分类效果。

然后用栅格搜索(grid search)结合五折交叉验证方法开始对模型进行训练。首先用一个较低的学习率找出最优的max_depth和min_child_weight组合。max_depth指的是每个弱分类树的最大深度，较小的值可以降低模型的复杂性，防止过拟合。min_child_weight是指每个叶节点上所有样本的权重之和的最小值。当某个叶节点上的样本权重之和小于min_child_weight时停止分裂。较大的min_child_weight值可以使模型更加保守，防止过拟合。训练结果如下图所示，最优参数组合为max_depth = 5, min_child_weight = 3。

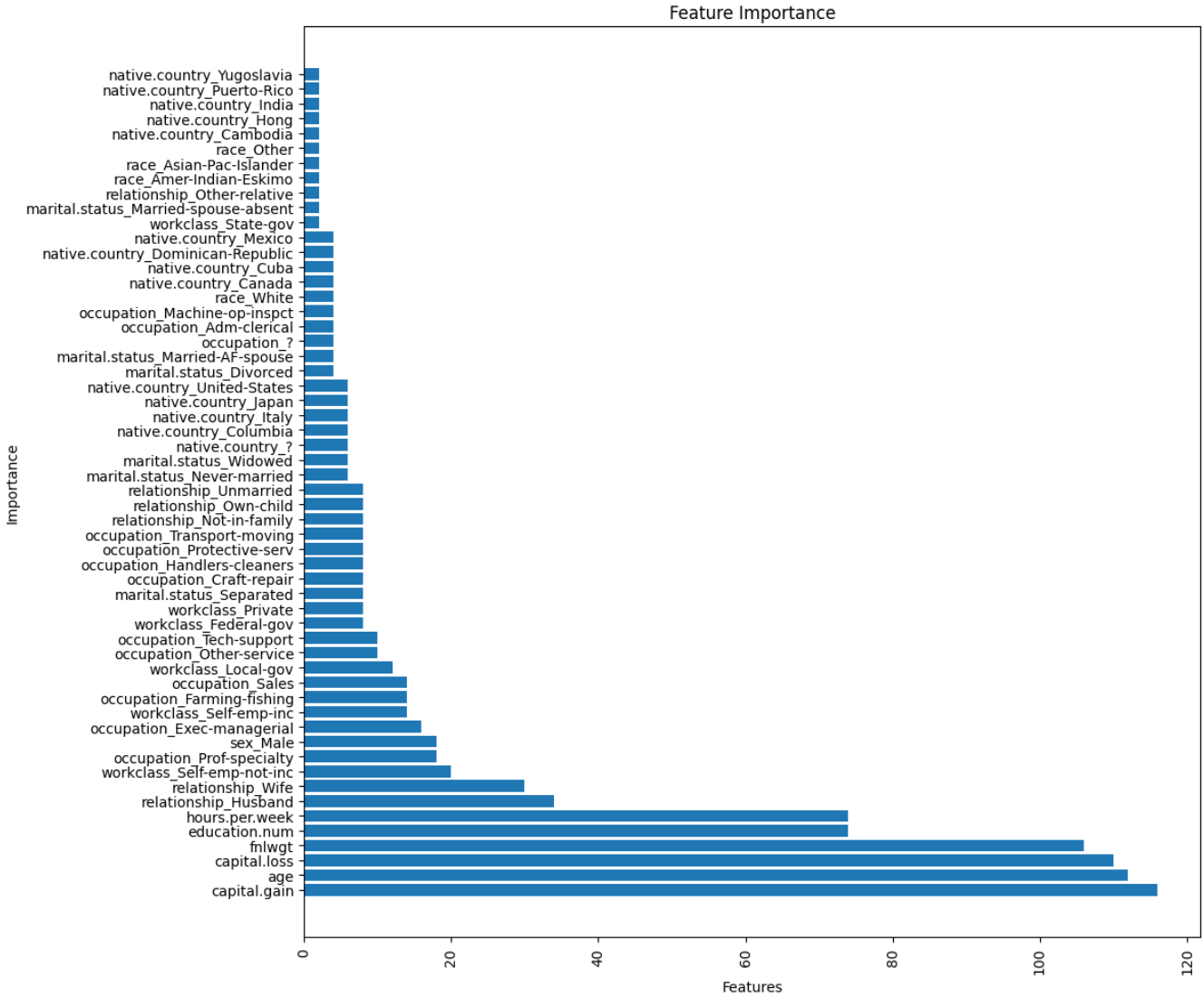


然后对参数gamma进行调整。gamma是一个正则化参数，用于控制决策树的叶节点分裂的条件。只有当分裂导致的损失函数的减少大于gamma时，才会执行分裂操作。较大的gamma值会使模型更加保守，防止过拟合。训练结果如下图所示，最优参数为gamma = 1.0。



至此，调参过程基本全部完成。将获得的最优参数给模型进行训练，得到预测准确率为0.8710243474446151。

因为XGBoost算法的弱分类器也是决策树，因此也可以让最终获得的模型输出属性的重要程度。其结果如下图所示。



3.3 K近邻

这一部分的代码参见KNN.ipynb文件。

3.3.1 方法

K近邻（K-Nearest Neighbors，简称KNN）算法是一种基本的分类和回归算法。它基于样本的特征值之间的距离进行分类或回归预测。KNN算法的基本思想是，对于给定的测试样本，根据其特征值与训练样本的距离，找出离它最近的K个样本，然后根据这K个邻居的标签（对于分类问题）或平均值（对于回归问题），来预测测试样本的标签或值。

KNN算法的主要步骤如下：

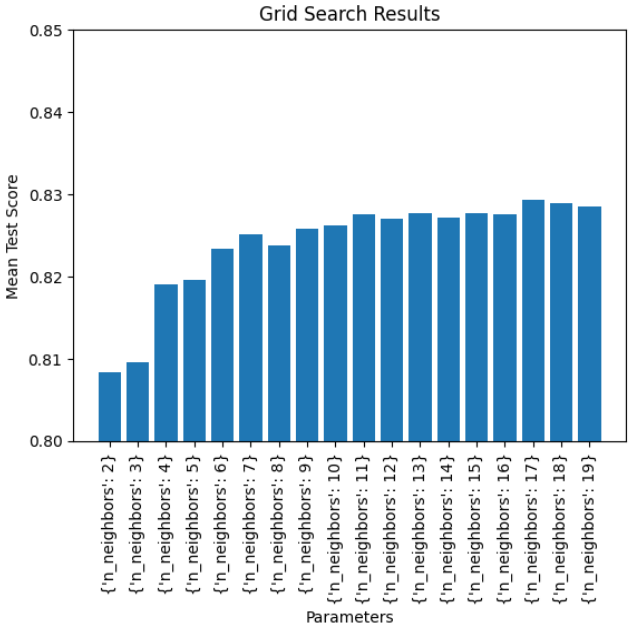
1. 计算测试样本与每个训练样本之间的距离，常用的距离度量方法有欧氏距离、曼哈顿距离等。
2. 根据距离的大小，选取离测试样本最近的K个训练样本。
3. 对于分类问题，根据K个邻居的标签，采取多数表决的方式确定测试样本的类别。
4. 对于回归问题，根据K个邻居的值，取平均值作为测试样本的预测值。
5. 输出预测结果。

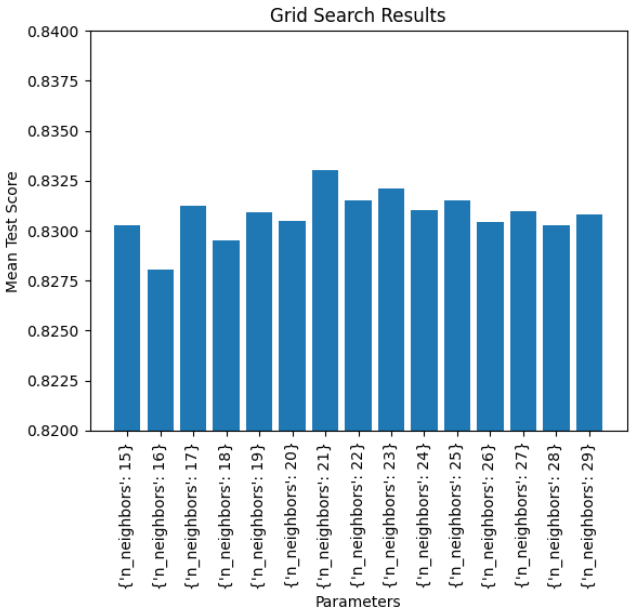
KNN算法的优点包括简单易实现、适用于多分类和回归问题、对异常值不敏感等。然而，KNN算法也有一些限制，例如计算复杂度高、对于高维数据和大规模数据集不适用、对于样本不平衡问题处理不佳等。

3.3.2 训练过程及结果

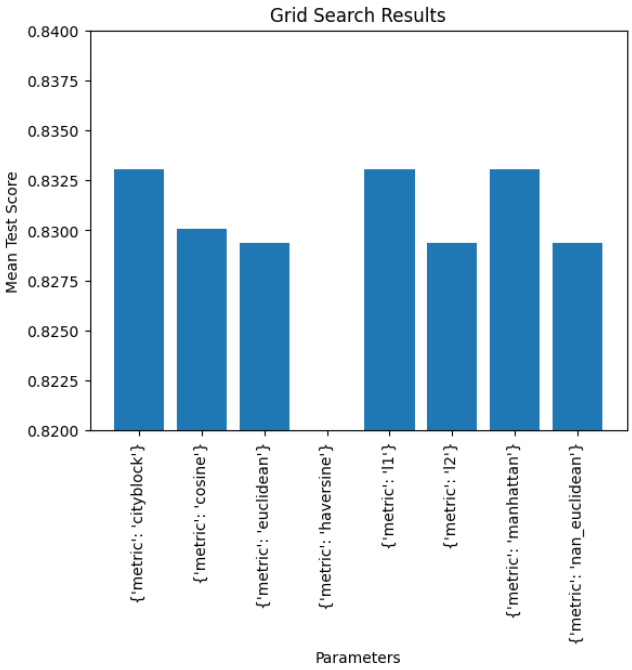
首先任意选取一组参数作为baseline。设置邻居数量为6，距离计算方式为曼哈顿距离（因为采用独热编码）训练模型，获得模型准确率为0.8313008104650249。

首先调整“邻居数量”参数，测试数据范围从2到19。训练结果如下图所示。可以看到，模型准确率基本上随着邻居数量增加而增长，拐点尚未出现，因此扩大数据范围进一步训练。





最终得到最优邻居数量为21，模型准确率为0.833049789869295。然后对计算距离方式进行调整，训练结果如下图所示，最优metric为cityblock，模型准确率为0.8355127608274948。



此外，还尝试了特征选择和降维、集成方法等方式进一步提高KNN算法的准确性，但效果都差不多，因此不再赘述。

3.4 神经网络

这一部分代码参见NN.ipynb文件。

3.4.1 方法

神经网络（Neural Network）是一种基于生物神经系统的模型，用于模拟和解决复杂的问题。它由大量的人工神经元（neuron）组成，这些神经元通过连接权重（weights）和激活函数（activation function）来进行信息传递和处理。

神经网络算法的基本思想是通过训练数据集来学习输入与输出之间的映射关系，从而实现对未知数据的预测和分类。算法通过调整连接权重，使网络能够对输入数据进行适应性学习，从而得出预测结果。

神经网络的基本组成部分包括输入层（input layer）、隐藏层（hidden layer）和输出层（output layer）。输入层接收输入数据，隐藏层通过一系列的线性变换和非线性激活函数对输入进行处理，最终输出层给出最终的预测结果。隐藏层的数量和大小可以根据问题的复杂程度和数据的特征进行灵活调整。

神经网络算法的训练过程通常包括以下步骤：

1. 初始化神经网络的连接权重和偏置项。
2. 将训练数据输入神经网络，并通过前向传播（forward propagation）计算预测结果。
3. 根据预测结果与真实标签之间的差异，使用损失函数（loss function）计算误差。
4. 通过反向传播（backpropagation）算法，将误差从输出层向前传播，并根据梯度下降法更新连接权重和偏置项，以减小误差。
5. 重复步骤2至步骤4，直到达到预定的停止条件（如达到最大迭代次数或误差降至阈值）。
6. 使用训练好的神经网络对新数据进行预测。

神经网络算法的优点包括能够处理复杂的非线性关系、具有较强的泛化能力和适应性，并且可以自动学习特征表示。然而，神经网络算法也存在一些挑战，如需要大量的训练数据和计算资源、模型的解释性较差等。

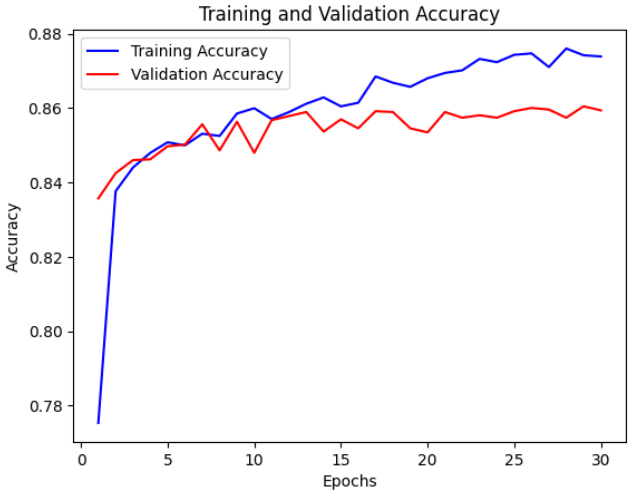
3.4.2 训练过程及成果

神经网络各种参数组合较多，调参过程此处只简单描述。

- 激活函数：输入层、隐藏层ReLu函数，在正区间上线性增长，在负区间上取值为0，有效缓解梯度消失问题；输出层softmax函数，将输入映射到多个类别的概率分布。
- 8层神经网络，隐藏层之间添加Dropout层，丢弃50%的神经元，防止过拟合。

- 隐藏层添加L2正则化参数防止过拟合，第一个隐藏层和最后一个隐藏层参数0.001，其余0.01，让中间层避免过拟合而输入输出层有更好学习效果。
- 优化器采用Adam（Adaptive Moment Estimation），结合了动量优化和自适应学习率的特性，可以有效地加快模型的收敛速度并提高训练效果。Adam优化器的核心思想是根据梯度的一阶矩估计（即梯度的平均）和二阶矩估计（即梯度的方差）来动态调整学习率。具体来说，Adam算法维护两个动态变量：梯度的一阶矩估计（即动量）和梯度的二阶矩估计。
- 定义自适应衰减回调函数，根据学习情况的好坏在迭代过程中递减学习率。

最后获得训练结果如下图所示。在测试集上准确率为 0.8593989910067997。



3.5 硬投票

集成学习中的投票机制是一种常用的集成策略，旨在通过结合多个模型的预测结果来提升整体的准确率。它利用了多个模型的独立性，通过集体决策来减少个别模型的错误和偏差，从而达到更好的性能。在硬投票（Hard Voting）中，每个模型都投票给一个类别，最终的预测结果由获得最多投票的类别决定。这种方式适用于二分类和多分类问题，可以通过简单的多数表决来确定最终的预测类别。

前四小节训练出的模型准确率如下：

模型名称	验证集准确率
决策树	0.8613731081377495
xgboost	0.8725597718797982
KNN	0.8355127608274948
NN	0.8530379469181838

该预测问题是一个社会科学和人口统计学问题，影响因素较多，较为复杂，因此模型准确率均在85%左右。然后对上述模型的预测结果通过硬投票的方法决定最终预测类别。

在投票过程中统计得到如下数据：

票数	数量
0	6996
4	1230
1	590
3	555
2	397

最终取票数为0和1的样本最终预测结果为0，其余为1。

4 分析总结

本次project中我使用了决策树、集成学习、K近邻和神经网络模型对样本个体的年收入是否超过$50K进行了预测。首先通过编码、归一化和特征筛选进行数据预处理。训练模型过程中使用网格搜索方法进行调参。训练完成后通过五折交叉验证来验证模型准确率。

本次训练得到的模型准确率均未超过90%，还有较大提升空间。后续可以尝试下面这些方法进一步提高模型准确率：

1. 数据预处理：更精细的数据清洗、缺失值处理、特征缩放等。还可以尝试进行特征工程，提取更有信息量的特征。
2. 模型选择和调参：尝试不同的模型算法，如随机森林和支持向量机等。
3. 数据扩增：通过数据增强技术（如旋转、翻转、缩放、裁剪等）扩充训练数据集，增加模型的泛化能力。

---

1. 通俗理解kaggle比赛大杀器xgboost, https://blog.csdn.net/v_JULY_v/article/details/81410574 ↩
2. https://zhuanlan.zhihu.com/p/29649128 ↩