華東師範大學

East China Normal University

数据挖掘论文

数据挖掘方法在收入预测中的应用

姓　　名：吴文韬

学　　号：10165000225

学　　院：统计学院

专　　业：统计学

指导教师：李艳

二〇一九 年　　五 月

目录

数据挖掘方法在收入预测中的应用

摘　要

在当前社会，社会分工和结构，居民的特性都可能与其收入之间有相关性，本文选用从美国人口普查局的数据库中提取的数据，对个人信息的特征进行提取从而预测其年收入是否会高于 5 万美元，从而可以划分不同收入阶层并总结相应的特征。针对这一问题，本文依据数据挖掘理论，基于 NaiveBayes、KNN、CART、RandomForest 和 Xgboost 五种算法对数据集建立模型，对数据集使用 Repeated hold-out 方法进行划分并训练测试，得到预测的准确率以及模型的一致性等指标并取均值，再结合每个模型的总运行时间进行模型的评价比较。结论是 Xgboost 算法下的模型最适于本文的数据，效果最优，准确率与 Kappa 分别达到 0.873 和 0.627，且运行时间也较短，相对的 NaiveBayes 理论下的模型拟合的效果最差。且 Xgboost 运行的结果显示，资本利得、年龄、婚姻状况和周工作时长等自变量对我们关心的收入变量的贡献度较高。

1　引言

1.1　选题动机

改革开放四十年来，随着我国经济的持续发展，综合国力显著增强，GDP 也保持着高速的增长，居民的生活水平得到了很大的提升。居民收入作为 GDP 增长的直观表现已成为经久不变的话题，也是我国人民需要关注的重大问题。在居民收入的来源中，工资性收入占居民收入的比重越来越大，已然成为了居民收入的重要组成部分，对居民收入的提高，居民收入差距的缩小，起到了关键的促进作用；但同样的，我们提出具有类似收入级别的居民是否存在共性特征的问题，这也是本文想要研究的问题。

1.2　问题介绍

众多文献都对居民收入的构成、收入的分配感兴趣，而对收入阶层划分下的特征并不明确，本文问题的出发点在于对已有的不同收入阶层的信息进行收集整理，以此提取出各个收入阶层的信息特征以及相应的重要性，本文所使用的数据将收入划分为两个阶层：年收入 5 万美元以上和年收入 5 万美元以下，并整理了其相应的个人信息，也就是我们想要提取特征的各个变量，依据这些变量的信息进行收入的分类，属于二分类（Binary classification）的预测问题。

2 方法介绍

2.1 NaiveBayes--朴素贝叶斯模型

朴素贝叶斯算法是一种基于概率统计的分类算法, 广泛应用于机器学习中分类问题的求解中。对分类任务来说，在所有相关概率都已知的理想情形下，贝叶斯决策论考虑如何基于这些概率和误判损失来选择最优的类别标记。贝叶斯判定准则：为最小化总体风险，只需在每个样本上选择那个能使条件风险 $R\left(c|\mathbf{x}\right)$ 最小的类别标记，即

$$h^{*}\left(\mathbf{x}\right)=\underset{c\in\varphi}{arg\,min}R\left(c|\mathbf{x}\right)$$

其中 φ 为可能的类别标记 $\varphi=\{c_1,c_2,...,c_N\}$，此时 h^{*} 称为贝叶斯最优分类器。在分类问题中我们的目的是最小化分类错误率，也就是说条件风险 $R\left(c|\mathbf{x}\right)=1-P\left(c|\mathbf{x}\right)$，因此我们需要获得后验概率 $P\left(c|\mathbf{x}\right)$，而后验概率可以根据贝叶斯公式得到，此处存在一个主要困难：类条件概率 $P\left(\mathbf{x}|c\right)$ 是所有属性上的联合概率。因此朴素贝叶斯分类器采用“属性条件独立性假设”：对已知类别，假设所有属性相互独立。

基于属性条件独立性假设，贝叶斯公式可写为：

$$P\left(c|\mathbf{x}\right)=\frac{P\left(c\right)P\left(\mathbf{x}|c\right)}{P\left(\mathbf{x}\right)}=\frac{P\left(c\right)}{P\left(\mathbf{x}\right)}\prod_{i=1}^{d}P\left(x_i|c\right)$$

其中 d 为属性条目，x_i 为 $\mathbf{x}$ 在第 i 个属性上的取值。

由于对所有类别来说 $P\left(\mathbf{x}\right)$ 相同，因此基于前式的贝叶斯判定准则有

$$h_{nb}\left(\mathbf{x}\right)=\underset{c\in\varphi}{arg\,max}\;P\left(c\right)\prod_{i=1}^{d}P\left(x_i|c\right)$$

这就是朴素贝叶斯分类器的表达式。

显然，朴素贝叶斯分类器的训练过程就是基于训练集 D 来估计类先验概率 $P\left(c\right)$，并为每个属性估计条件概率 $P\left(x_i|c\right)$。

2.2 KNN

k 近邻 (k-Nearest Neighbor，简称 kNN) 学习是一种常用的监督学习方法，所谓 k 最近邻，就是 k 个最近的邻居的意思，说的是每个样本都可以用它最接近的 k 个邻居来代表。该方法的思路是：如果一个样本在特征空间中的 k 个最相似 (即特征空间中最邻近) 的样本中的大多数属于某一个类别，则该样本也属于这个类别。其工作机制是：给定测试样本，基于某种距离度量找出训练集中与其最靠近的 k 个临近样本，然后基于这 k 个邻居的信息来进行预测。

Algorithm 1 KNN

1: 准备数据，对数据进行预处理
2: 选用合适的数据结构存储训练数据和测试元组
3: 设定参数，如 k
4: 维护一个大小为 k 的的按距离由大到小的优先级队列，用于存储最近邻训练元组。随机从训练元组中选取 k 个元组作为初始的最近邻元组，分别计算测试元组到这 k 个元组的距离，将训练元组标号和距离存入优先级队列
5: 遍历训练元组集，计算当前训练元组与测试元组的距离，将所得距离 L 与优先级队列中的最大距离 $Lmax$
6: 进行比较。若 $L >= Lmax$，则舍弃该元组，遍历下一个元组。若 $L < Lmax$，删除优先级队列中最大距离的元组，将当前训练元组存入优先级队列。
7: 遍历完毕，计算优先级队列中 k 个元组的多数类，并将其作为测试元组的类别。
8: 测试元组集测试完毕后计算误差率，继续设定不同的 k 值重新进行训练，最后取误差率最小的 k 值。

2.3 CART 决策树

决策树是一类常见的机器学习算法。顾名思义，决策树是基于树结构来进行决策的。一般的，一颗决策树包含一个根节点、若干个内部节点和若干个叶节点；叶节点对应决策结果，其他每个节点对应一个属性测试；每个阶段包含的样本集合根据属性测试的结果被划分到子节点中。由此可以看到决策树学习的关键在于安排最适合的属性测试，也就是每一次的划分所依据的属性是要最优的。CART(Classification And Regression Tree) 是 Breiman 等人在 1984 年提出的一种二分决策树，使用基尼系数 (Gini index) 来选择最优划分属性。

Algorithm 2 CART

Input: 训练数据集 D，停止条件

Output: $CART$ 决策树 T

1: 设结点的训练数据集为 D，计算现有特征对该数据集的 $Gini$ 系数。此时，对每一个特征 A，对其可能取的每个值 a，根据样本点对 $A = a$ 的测试为“是”或“否”将 D 分割成 $D1$ 和 $D2$ 两部分，计算 $A = a$ 时的 $Gini$ 系数：

$$Gini\left(D, A\right) = \frac{|D_1|}{D} GiniD_1 + \frac{|D_2|}{D} GiniD_2$$

2: 在所有可能的特征 A 以及它们所有可能的切分点 a 中，选择 $Gini$ 系数最小的特征及其对应的切分点作为最优特征与最优切分点。依最优特征与最优切分点，从现结点生成两个子结点，将训练数据集依特征分配到两个子结点中去。

3: 对两个子结点递归地调用步骤 $1 \sim 2$，直至满足停止条件。

4: 生成 $CART$ 决策树 T。

2.4 RandomForest

随机森林 (Random Forest) 就是通过集成学习的思想将多棵树集成的一种算法，它的基本单元是决策树。RF 在以决策树为基学习器构建 Bagging 集成的基础上，进一步在决策树的训练过程中引入了随机属性选择。具体来说，传统决策树在选择划分属性时是在当前结点的属性集合中选择一个最优属性，而在 RF 中，对基决策树的每个节点，先从该节点的属性集合中随机选择一个包含 k 个属性子集，然后再从这个子集中选择一个最优属性用于划分。随机森林的方法由于有了 bagging，也就是集成的思想在，实际上相当于对于样本和特征都进行了采样（如果把训练数据看成矩阵，就像实际中常见的那样，那么就是一个行和列都进行采样的过程），所以可以避免过拟合。

随机森林在 bagging 的基础上更进一步：

1) 从样本集中用 Bootstrap(有放回的随机抽样) 随机选取 n 个样本
2) 从所有属性中随机选取 K 个属性，选择最佳分割属性作为节点建立 CART 决策树（这里面也可以是其他类型的分类器）
3) 重复以上两步 m 次，即建立了 m 棵 CART 决策树
4) 这 m 个 CART 形成随机森林，通过投票表决结果，决定数据属于哪一类。

2.5 Xgboost

Xgboost 是“eXtreme Gradient Boosting”的简称，即极端梯度提升树，是在 GBDT 基础上的改进算法，由 Chen 等于 2015 年提出。Gradient Boosting 属于集成算法的 Boosting 方法中的一种类别，Boosting 分类器属于集成学习模型，其基本思想是把成百上千个分类准确率较低的树模型组合成一

个准确率较高的模型。Gradient Boosting 则是通过加入新的弱学习器来改善已有弱学习器的残差，将多个学习器相加起来得到最终的预测结果，具体是在生成每一棵树的时候采用梯度下降的思想，以上一步生成的所有树为基础，向着最小化给定目标函数的方向前进。XGBoost 算法作为 GBDT 算法的改进版本，相比 GBDT 在优化时只用到一阶导数信息，XGBoost 则对损失函数进行了二阶泰勒展开，充分利用了一阶和二阶导数，并在损失函数之外对正则项求得最优解。且 Xgboost 能自动利用 CPU 的多线程进行并行，并对算法加以改进以提高精度。Xgboost 的目标函数：

$$Obj = \sum_i l\left(\hat{y}_i, y_i\right) + \sum_k \Omega\left(f_k\right)$$

其中

$$\Omega\left(f\right) = \gamma T + \frac{1}{2}\lambda \left\|w\right\|^2$$

$l\left(\hat{y}_i, y_i\right)$ 为损失函数，用来度量预测 $\hat{y}_i$ 和目标 y_i 之间的差异；$\sum_k \Omega\left(f_k\right)$ 为树的复杂度，Ω 也被称为正则惩罚项，与叶子节点的数量 T 和叶子节点的值有关。额外的正则化项有助于平滑最终学习的权重，避免过度拟合。

上公式的集成决策树模型中的目标函数采用 Additive Training(Boosting) 方法训练，即从常数预测开始，每次添加一个新函数:

$$\begin{gathered}
\hat{y}_i{}^{(0)} = 0 \\
\hat{y}_i{}^{(1)} = f_1\left(x_i\right) = \hat{y}_i{}^{(0)} + f_1\left(x_i\right) \\
\hat{y}_i{}^{(2)} = f_1\left(x_i\right) + f_2\left(x_i\right) = \hat{y}_i{}^{(1)} + f_2\left(x_i\right) \\
\cdots \\
\hat{y}_i{}^{(t)} = \sum_{k=1}^{t} f_k\left(x_i\right) = \hat{y}_i{}^{(t-1)} + f_t\left(x_i\right)
\end{gathered}$$

传统的 GBDT 在求解的过程中只利用了一阶导数的信息，而 Xgboost 对损失函数进行二阶泰勒展开（第 t 次迭代的目标函数）：

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

其中 $g_i = \partial_{\hat{y}^{(t-1)}} l\left(y_i, \hat{y}^{(t-1)}\right)$ 和 $h_i = \partial^2_{\hat{y}^{(t-1)}} l\left(y_i, \hat{y}^{(t-1)}\right)$ 是关于损失函数的一阶和二阶梯度统计量。我们可以去掉常数项，得到步骤 t 的简化目标。

$$\bar{Obj}^{(t-1)} \simeq \sum_{i=1}^{n}\left[g_i f_t\left(x_i\right) + \frac{1}{2}h_i f_t^2\left(x_i\right)\right] + \Omega\left(f_t\right)$$

3 数据介绍

3.1 数据来源

本文所选用的数据集是从美国人口普查局的数据库[1]中提取的，是 Ron Kohavi 在 1996 年发表的利用决策树提高贝叶斯分类器的精度的论文中所使用的数据集，由 Barry Becker 从 1994 年的人口普查数据库中进行的提取。我们使用 R 语言来对该数据集进行处理建模，利用`dim()`函数可以看到该数据集有 32561 个观测，15 个变量，其中的 income 是我们所关心的变量，表示这条观测所对应的人的年收入是否超过 5 万美元。

3.2 变量介绍

所有变量所代表的含义解释如下表所示：

表 1: 变量解释

变量名	含义解释	变量类型
age	年龄	数值型
workclass	工作性质	因子型 (9 个 level)
fnlwgt	最终权重	数值型
education	受教育水平 (学历)	因子型 (16 个 level)
education-num	受教育年限	数值型
marital-status	婚姻状况	因子型 (7 个 level)
occupation	职业	因子型 (15 个 level)
relationship	家庭身份	因子型 (6 个 level)
race	人种	因子型 (5 个 level)
sex	性别	因子型 (2 个 level)
capital-gain	资本利得	数值型
capital-loss	资本损失	数值型
hours-per-week	每周工作时长	数值型
native-country	国籍	因子型 (42 个 level)
income*	收入状况	因子型 (2 个 level)

* 所标注的变量为关心变量

fnlwgt(final weight): 是人口普查认为该条目代表的人数。

[1] http://www.census.gov/ftp/pub/DES/www/welcome.html

3.3 描述性统计

3.3.1 数值型变量的 summary

表 2: 各变量统计描述

变量名	均值	标准差	中位数	截尾均值	绝对中位差	最小值	最大值	极差	偏度	峰度	下四分位数	上四分位数
age	38.58	13.64	37	37.69	14.83	17	90	73	0.56	-0.17	28	48
fnlwgt	189778.37	105549.98	178356	180802.36	88798.84	12285	1484705	1472420	1.45	6.22	117827	237051
education.num	10.08	2.57	10	10.19	1.48	1	16	15	-0.31	0.62	9	12
capital.gain	1077.65	7385.29	0	0	0	0	99999	99999	11.95	154.77	0	0
capital.loss	87.3	402.96	0	0	0	0	4356	4356	4.59	20.37	0	0
hours.per.week	40.44	12.35	40	40.55	4.45	1	99	98	0.23	2.92	40	45

截尾均值为去掉首尾极端值后的均值；绝对中位差 (MAD) 为数据到中位数的偏差绝对值的中位数

此表由 psych 包中的 describe() 函数所得

3.3.2 数值型变量的相关性

除了上述的描述，本文还绘制了数值型变量的相关性图，如下图所示：

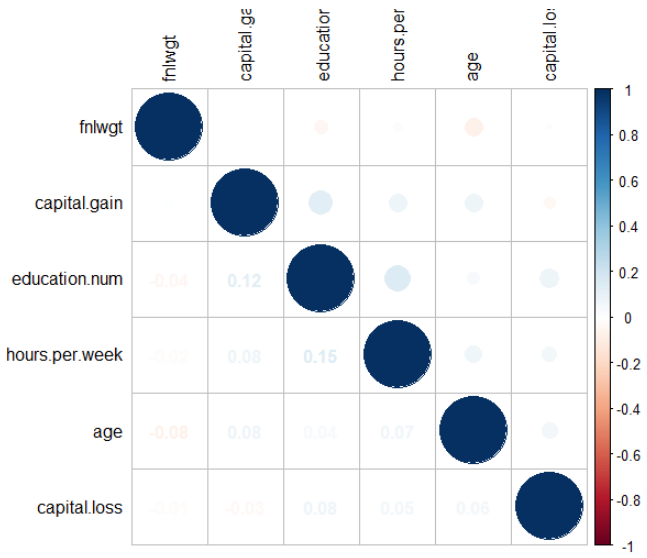

图 1: 数值型变量的相关性可视化 (Corrplot)

从图中我们可以看到，六个数值型变量之间并没有很强的相关性。

3.3.3 因子型变量的 summary

针对数据集中的 9 个因子型变量 (包含关心变量 income)，我们对 level 数较多的变量绘制帕累托图 (Pareto chart，是按照发生频率大小顺序绘制的直方图)，而对于 level 数较少的变量 (sex,income) 只绘制简单的柱状图。如下：

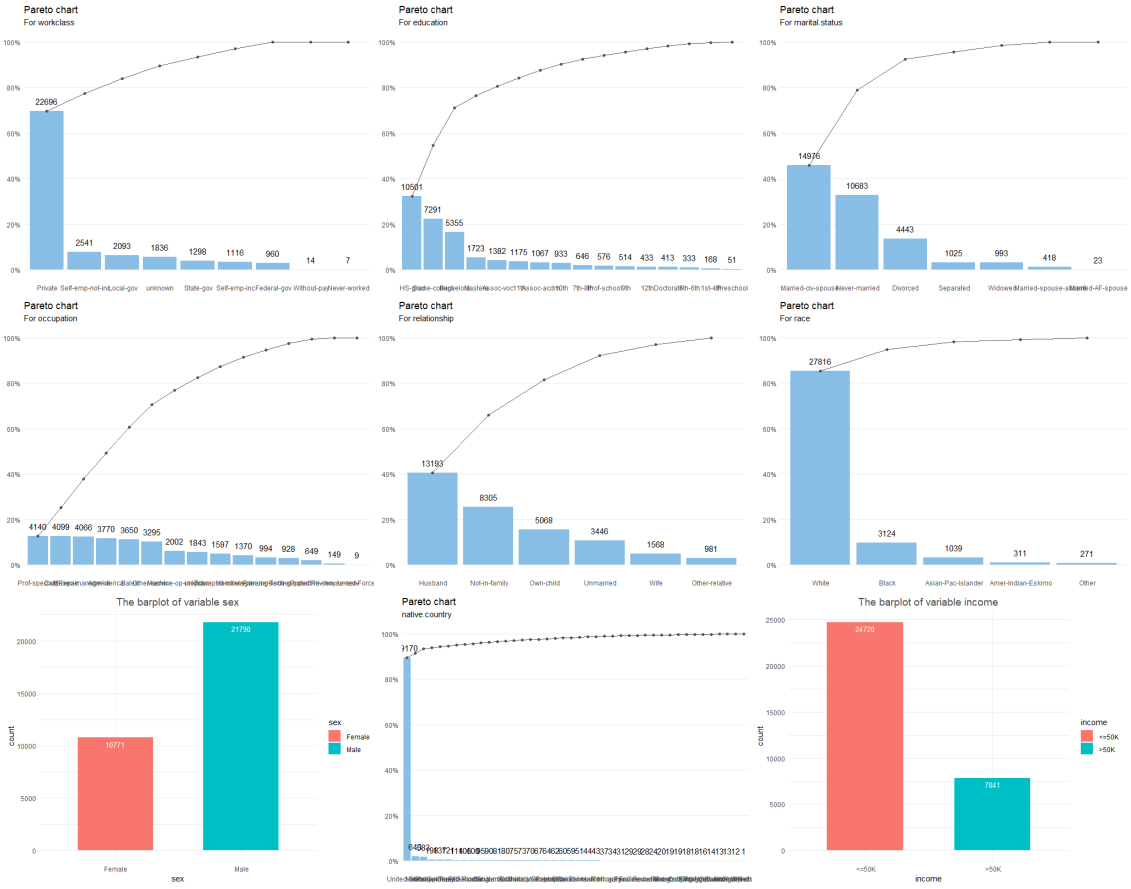

图 2: 因子型变量的帕累托图与条形图

3.4 预处理

有了上文对数据集的描述，在建立模型之前，我们需要先对数据集进行预处理。首先利用 R 中的`sum(is.na())`函数得出此数据集无缺失值；根据上文的内容，我们可以看出 fnlwgt 这个数值型变量本身的值过大，因而得到的标准差也过大，考虑其实际含义，本文决定将此变量剔除；而对于 education 与 education.num 这两个变量，education.num 表示受调查者受教育的年数，在一定程度上与 education 这个变量是对应的，考虑到我们将会用 Xgboost 等方法，本文决定保留因子型变量的 education 变量，以适用于后面的独热编码处理。

本文的关心变量 income 有两个 level，年收入小于等于 5 万美元与大于 5 万美元，其数量分别为 24720 与 7841，我们在 R 语言中将 level 名称重命名为 1 和 2，并运用 caret 包中的`createDataPartition()`函数来进行训练集与测试集的划分，保证训练集与测试集中关心变量 income 两个 level 的比例与原数据集中相似。使用 Repeated hold-out 方法重复 2 次，设置随机种子为 1 和 2，取 80% 为训练集，20% 为测试集，取出的数据集数据量如下表所示。（后续的的模型评价时是对两次划分分别处理并取均值）

表 3: 训练 | 测试集数据情况

数据集	<=50K(1)	>50K(2)	总和
训练集	19776	6273	26049
测试集	4944	1568	6512
原数据集	24720	7841	32561

4 数据分析

4.1 NaiveBayes 建模

使用 R 语言中的`klaR`包中的`NaiveBayes()`函数来进行朴素贝叶斯的建模，在简单的朴素贝叶斯建模中，需要寻找的最优参数为`usekernel=FALSE or TRUE`，表示是否使用核密度估计，如果为 TRUE，则使用核密度估计 (density) 进行密度估计。如果为假，则用正态 (normal) 密度进行估计；以及`fL=0 or 1`，表示是否用拉普拉斯修正，如果为 0 则不用，如果为 1 则表示使用。本文使用了留一交叉验证的方法来寻找这两个参数的优秀解，并将取不同参数时的错误率整理成表格如下：

表 4: 留一交叉验证不同参数下的错误率

usekernel \ fL	0	1
TRUE	0.1779339	0.1779339
FALSE	0.1817728	0.1817728

由上表可知最优参数取 usekernel = TRUE，fL = 0 时错误率最小，使用最优参数对训练集进行朴素贝叶斯建模，并使用模型对测试进行预测，将得到的预测结果与测试集关心变量的真实值进行对比，利用`caret`包中的`confusionMatrix()`函数整理出混淆矩阵 (confusionmatrix)，如下表所示：

表 5: 基于最优参数的朴素贝叶斯模型预测的混淆矩阵

预测类别 \ 真实类别	1	2	总计
1	4861	1111	5972
2	83	457	540
总计	4944	1568	6512

1 表示 income<=5K，2 表示 income>5K

4.2 KNN 建模

使用 R 语言`kknn`包中的函数进行基于 KNN 算法的建模，在 kknn() 函数中，我们需要优化的参数是`kernel`和`k`，分别代表加权方法和寻找的最近邻居个数。在前文我们提到，KNN 算法是基于依靠某种距离度量所找到的 k 个邻居的信息来进行预测，因此我们需要知道到底利用周围多少邻居的信息才合适，同时我们也需要知道以何种权重来利用这些邻居的信息，通常默认的方法是使用等权重的 kernel，也就是每个邻居给预测样本的影响相同，那么在所有邻居的预测结果中就遵从少数服从多数，当然在选择其他 kernel 后就有对每个邻居的信息采用不同的赋权方法。于是我们使用`train.kknn()`函数，基于留一交叉来判断这两个参数的最优取值。在这个函数中现有的 kernel 选择为"rectangular"(也就是等权重),"triangular","epanechnikov"(or beta(2,2)),"biweight"(or beta(3,3)),"triweight"(or beta(4,4)),"cos","inv","gaussian","rank"and"optimal"；对于邻居数量的选择通常不会很高，所以在`train.kknn()`函数中设置 k 的范围最大不会超过 50。运行后得到的错误率表格及图像如下：

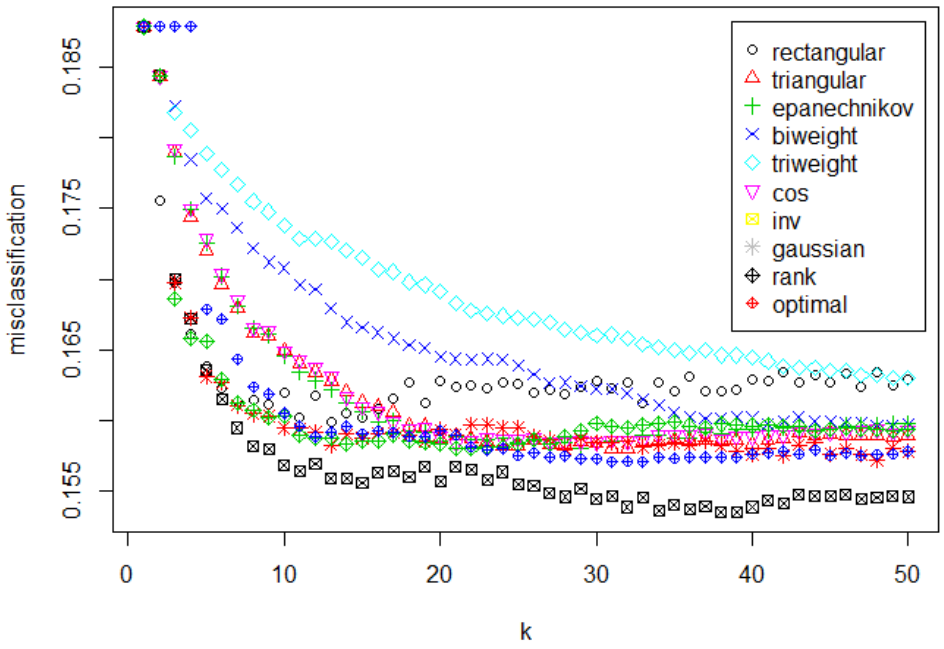

图 3: 留一交叉验证下不同参数的错误率

观察上图，我们发现模型的错误率随着 k 的增大而减小，在大约 k>15 以后错误率趋于平稳。通过运行结果可以得到最优参数取值为 kernel = inv，k = 38。使用最优参数对训练集进行 KNN 算法下的建模，并预测测试集，得到的混淆矩阵如下:

表 6: 基于最优参数的 KNN 模型预测的混淆矩阵

预测类别 \ 真实类别	1	2	总计
1	4576	639	5215
2	368	929	1297
总计	4944	1568	6512

1 表示 income<=5K，2 表示 income>5K

4.3 CART 建模

使用 R 语言`rpart`包中的`rpart()`函数进行基于 CART 算法下的决策树建模，需要设置的参数有 cp，minsplit 和 maxdepth 等。cp 全称为 complexity parameter，指某个点的复杂度，对每一步拆分，模型的拟合优度必须提高的程度，这个参数的主要作用是通过删除明显不值得的分割来节省计算时间，cp 的取值取决于 xerror。minsplit 表示一个节点中必须存在的最小观测值，以便尝试分裂，本文取 minsplit = 1。maxdepth 为设置最终树的任意节点的最大深度，也就是树的最大层数由于训练集中总共有 12 个自变量，因此取 maxdepth = 12。为了防止过拟合，本文对 rpart 函数得到的决策树进行剪枝，基于 1-SE 准则，绘制了 cp 与相对错误率 xerror 的关系图如下：

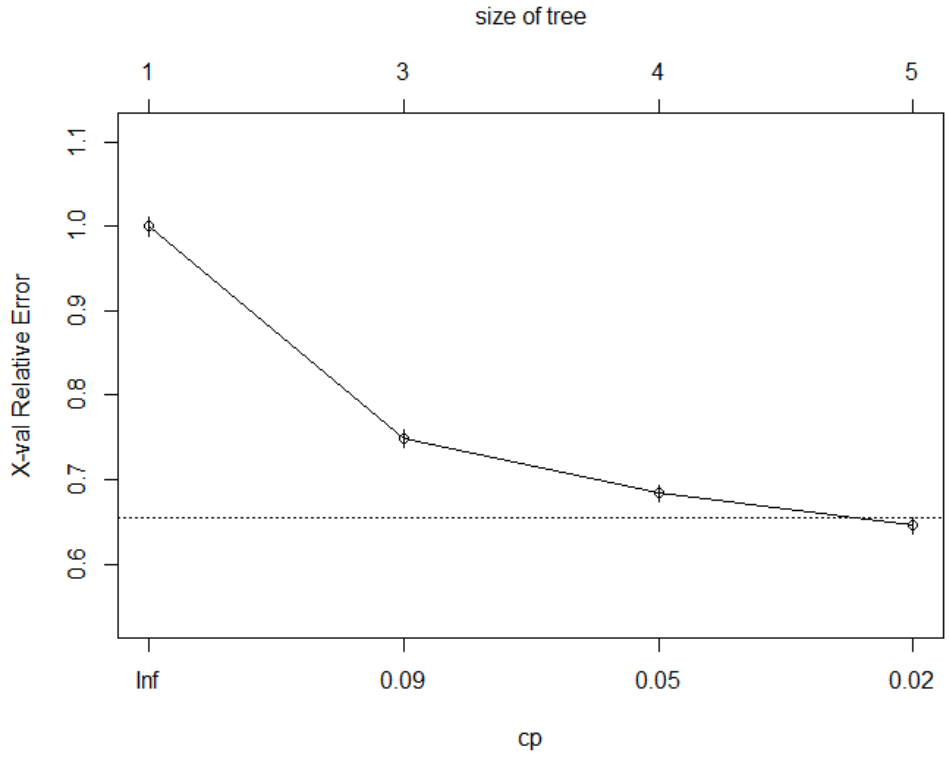

图 4: cp 与 xerror 关系图

由图及 1-SE 准则，我们可以认为 cp 值应该取 0.01，基于这样的 cp 值，我们最终得到的决策树为

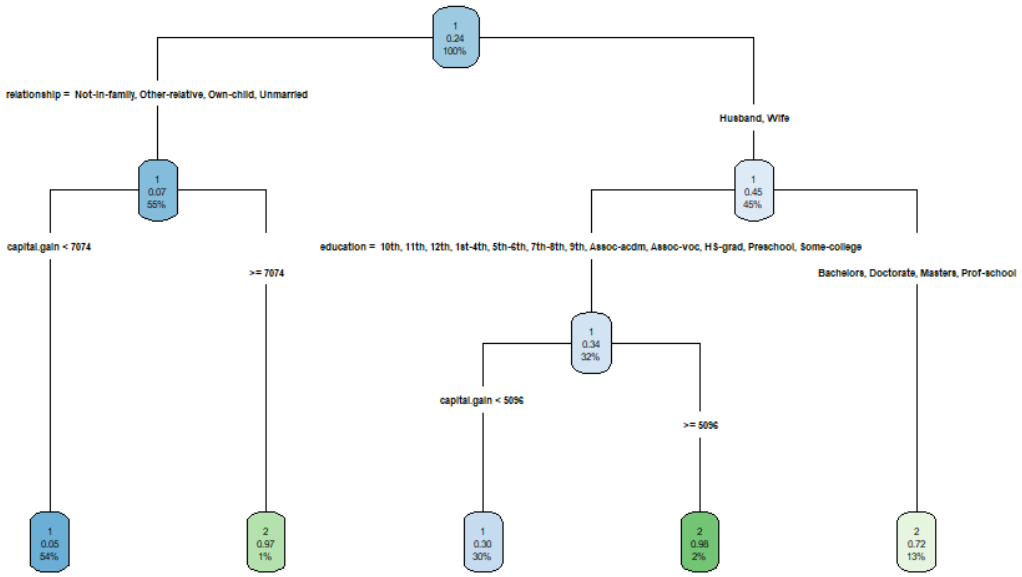

图 5: CART 算法下的决策树

基于此决策树，我们对测试集进行预测，得到的混淆矩阵如下：

表 7: 基于 CART 的决策树模型预测的混淆矩阵

预测类别 \ 真实类别	1	2	总计
1	4679	750	5429
2	265	818	1083
总计	4944	1568	6512

1 表示 income<=5K，2 表示 income>5K

变量的相对重要性：

表 8: 变量相对重要性

relationship	marital.status	capital.gain	education	sex	occupation
1901.40037	1869.63781	842.36637	747.76250	594.45969	535.21331
age	hours.per.week	native.country	capital.loss	race	
436.13059	245.71646	17.48404	15.56506	2.34542	

4.4 RF 建模

使用 R 语言中的`randomForest`包来进行随机森林的建模。需要设置的参数有很多，本文仅选取了其中几个参数进行优化。

* ntree：在森林中树的个数。默认值为 500，本文取使错误率最小的 n，经过循环得到 ntree = 234.
* mtry：每棵树使用的特征个数，即在每次分割时随机抽样作为候选变量的变量数。本文取使错误率最小的 m，经循环比较得到 mtry = 2.
* importance：是否计算变量的特征重要性，默认为 False。本文需要计算变量的重要性，因此取 TRUE。
* nodesize：终端节点的最小尺寸，相当于 minsplit，故本文取 nodesize = 1.
* maxnodes：在森林中可以有最多的终端节点树，相当于 maxdepth，在二分类问题中每层的节点数等于 2 的层数次方，故本文取 maxnodes = 2^12

其中 ntree 与 mtry 的选择可由下图参考：

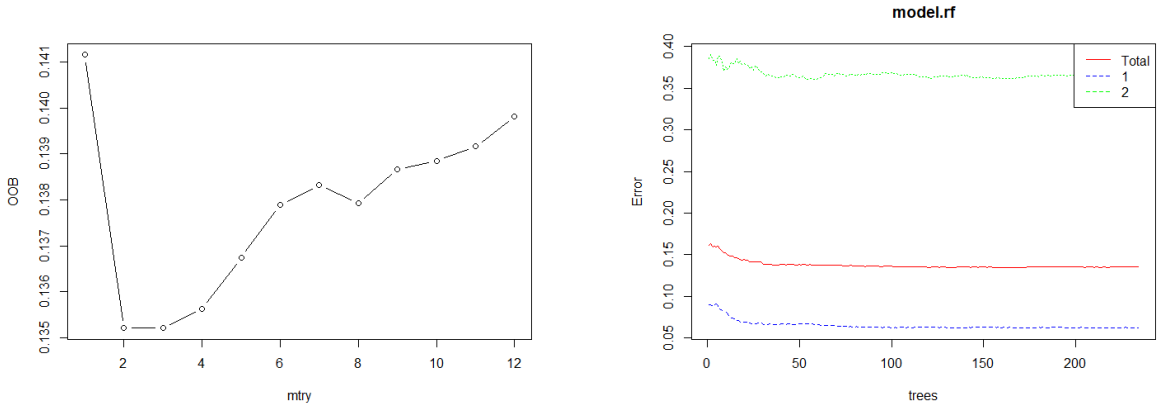

图 6: 不同 ntree 与 mtry 取值下的袋外错误率 OOB

通过对上图的观察，我们可以看到在 mtry = 2 时 OOB 取值最小，故此为最优参数；而在 mtry = 2 下的 ntree 取值大约在 234 时 OOB 最小，因此此为最优参数。

基于上述选择的参数，我们使用`randomForest()`函数进行建模并对测试集进行预测，得到的混淆矩阵如下：

表 9: 基于 RandomForest 模型预测的混淆矩阵

预测类别 \ 真实类别	1	2	总计
1	4660	547	5207
2	284	1021	1305
总计	4944	1568	6512

1 表示 income<=5K，2 表示 income>5K

同时得到的变量重要性 (variable importance) 如下图所示：

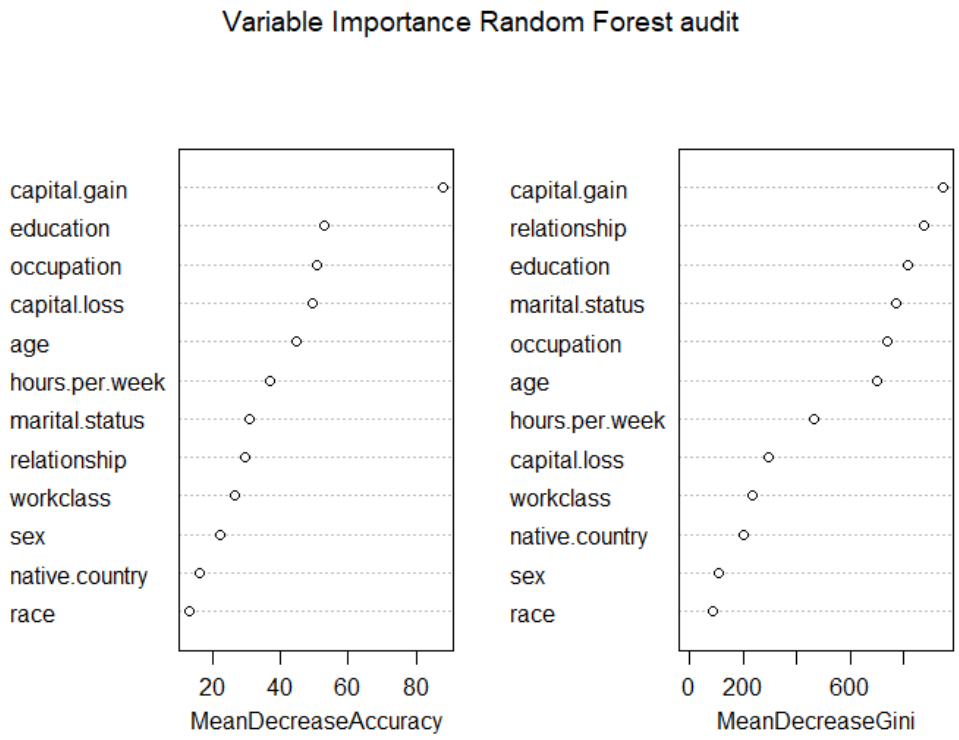

图 7: RandomForest 下的变量重要性

由上图可知，这里的变量重要性度量分别采用了“正确率平均减小量”和“Gini 指标平均减小量”，这两个量都可以衡量变量对于我们关心变量的贡献度。可以看出两种指标下 capital.gain 的贡献均最高，贡献程度较高的还有 education、occupation、relationship 和 marital.status，而 race、sex、workclass 和 native.country 变量的贡献在这两种指标下均较低，这与前文 CART 算法下得到的重要变量较为类似。

4.5 Xgboost 建模

本文最后使用的模型是基于 Xgboost，利用 R 语言的`xgboost`包进行建模。需要设置的参数同样很多，因为作者对于 Xgboost 的了解并不深以及时间因素，无法做到准确的函数的参数优化，此处所取的参数均为结合通常情况下的经验和在 R 语言中设置不同参数运行下得到的综合结果。由于本文涉及问题为二分类问题，且自变量中存在因子型变量，因此先对训练集与测试集进行独热编码。下表为 Xgboost 中需要优化的部分参数：

* booster : 选择每次迭代的模型，有两种选择：gbtree 基于树的模型 & gbliner 线性模型. 本文选择默认参数 gbtree.
* nrounds : 最大迭代次数，没有默认值，随着迭代次数的增加判定的准确率会近似到某个值，因此本文取相对较大值 100
* min_child_weight : 决定最小叶子节点样本权重和，这个参数用于避免过拟合。当它的值较大时，可以避免模型学习到局部的特殊样本；但是如果这个值过高，会导致欠拟合。本文取 0.1.
* max_depth : 树的最大深度，越大则树模型越复杂、越容易过拟合，本文取值 10.
* subsample : 这个参数控制对于每棵树随机采样的比例。减小这个参数的值，算法可以避免过拟合。但如果这个值设置得过小可能会导致欠拟合。本文取默认值 1.
* colsample_bytree : 用来控制每棵随机采样的列数的占比 (每一列是一个特征)，越大则计算越耗时、树模型精度越高，但可能导致过拟合。本文取 0.3.
* gamma : 指定节点分裂所需的最小损失函数下降值，本文取 5.
* objective : 任务目标，因为本文所研究的是二分类，故设置为“binary:logistic”

根据上述选取的各参数值进行建模，并对测试集进行预测，得到的混淆矩阵如下：

表 10: 基于 Xgboost 模型预测的混淆矩阵

预测类别 \ 真实类别	1	2	总计
1	4685	570	5255
2	259	998	1257
总计	4944	1568	6512

1 表示 income<=5K，2 表示 income>5K

同时得到的变量重要性 (variable importance) 如下图所示：

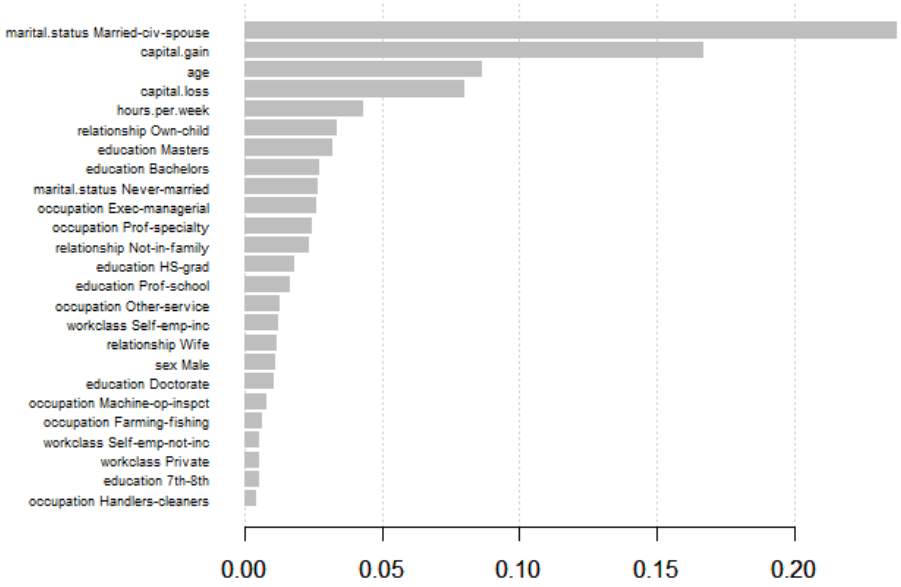

图 8: Xgboost 下的变量重要性

从图中也能隐约看到 marital.status 的贡献最高，capital.gain、age、capital.loss 和 hours.per.week 的贡献也较高，这与前文 CART 和 RandomForest 的结论也有相似之处。

4.6 模型效果比较

在前面我们分别用 NaiveBayes、KNN、CART、RandomForest 和 Xgboost 五种算法对数据进行建模预测，通过在测试集上预测所得到的混淆矩阵，使用 Repeated hold-out 方法进行重复 2 次（设定不同的随机种子进行划分，分别进行建模预测并取结果的均值），取均值整理出五种方法下的各指标值：

表 11: 五种模型所得混淆矩阵的指标比较

模型 / 指标	NaiveBayes	KNN	CART	RandomForest	Xgboost
准确率（Accuracy）	0.817	0.845	0.844	0.872	0.873
精确率（Precision）	0.846	0.716	0.755	0.782	0.794
灵敏度（Sensitivity）	0.291	0.593	0.522	0.651	0.637
特异度（Specificity）	0.983	0.926	0.946	0.943	0.948
F1 Score	0.434	0.649	0.617	0.711	0.707
Kappa	0.354	0.551	0.523	0.630	0.627
AUC	0.892	0.876	0.847	0.907	0.930

同时我们还根据预测情况绘制了五种模型下的各指标图像：

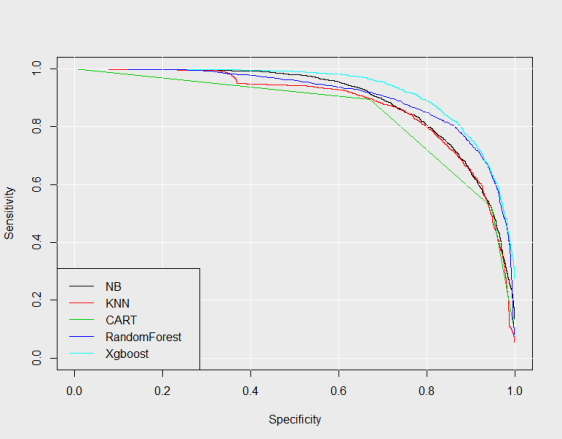

图 9: 敏感度特异度曲线

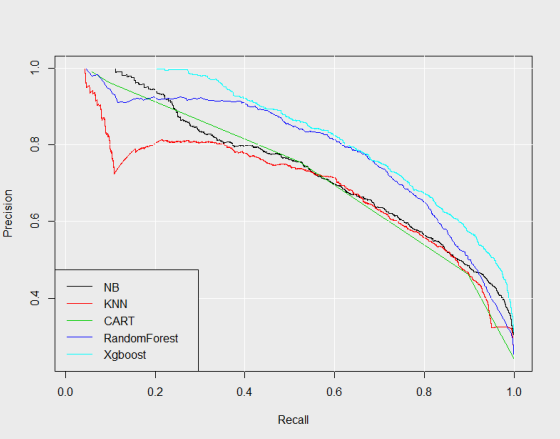

图 10: 召回率精确率曲线

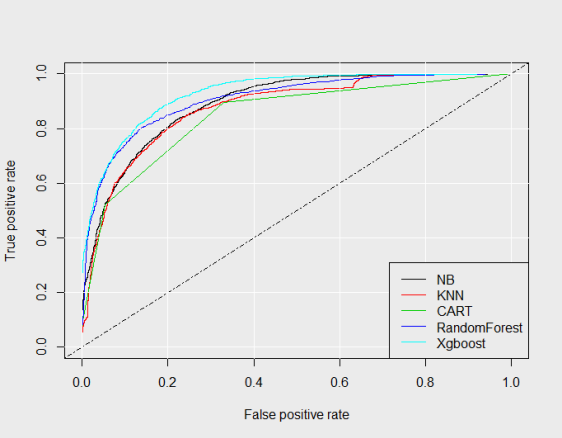

图 11: ROC

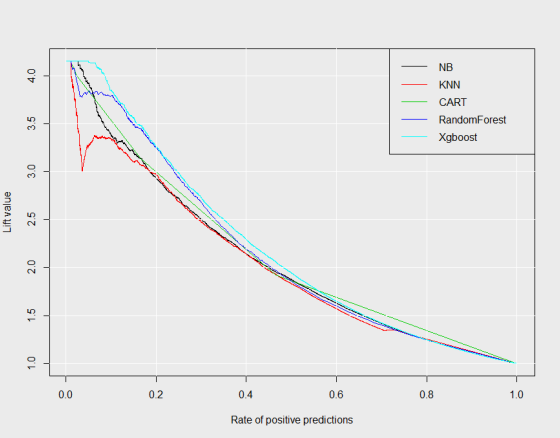

图 12: 提升图

图 13: 四种模型评价曲线

通过观察指标表，我们可以看到，从 Kappa 值的角度来说，RandomForest 模型的 Kappa 值最高，Xgboost 的 Kappa 值与其近似，可以认为这两种模型的效果一致性较好，而 NaiveBayes 的 Kappa 值最低，也就是说模型的效果最差，总体上来讲，除过 NB 模型，其余四个模型的 Kappa 值均在可接受范围内。从准确度的角度来说，同样是 RandomForest 与 Xgboost 模型的预测最准确，预测出错的可能性相对较小，NaiveBayes 模型的预测准确率最低，但也在可接受的范围内；从 F1 score 的角度来看，RandomForest 与 Xgboost 模型的值较高，表示其输出结果由于其他三个模型，NaiveBayes 的 F1 值最低，输出结果较差；从 AUC 的角度来看，Xgboost 的 AUC 值最高，RandomForest 其次，CART 模型的 AUC 值最小，但五种模型的 AUC 均高于 0.5，表示均优于随机分类器，而在这五种模型中 Xgboost 分类器的分类效果最好。

观察指标线，从 ROC 曲线图中我们可以看到，五种模型的曲线均在对角线之上，说明效果均优于随机分类器，且显然 Xgboost 的 ROC 曲线要在其余四种模型的曲线之上，说明 Xgboost 模型的效果要优于前四种模型，但其实 Xgboost 模型的效果与 RandomForest 模型的效果相差无几；四幅图中我们都可以看到 Xgboost 模型对应的曲线要高于其余四条曲线，也就是从整体上来讲本文中所使用的 Xgboost 模型最优。

综合表和图像的结果，本文认为 Xgboost 模型的效果最优，RandomForest 其次，但与 Xgboost 相差不大，其他三种模型效果明显不如这两种，可以认为 NaiveBayes 模型的效果最差，KNN 与 CART 决策树模型的效果一般。另外，本文在建立 Xgboost 模型时并没有做严格意义上的参数优化，参数选取仅仅是查阅文献与反复试验所得到的相对最优结果，因此可以说 Xgboost 模型的参数仍有进一步优化的可能。

最后，本文使用 R 语言中的`proc.time()`函数记录了五种模型的运算时间，包括参数优化、建模和预测三部分的时间总和，其结果如下表：

表 12: 模型运行时间

模型	NaiveBayes	KNN	CART	RandomForest	Xgboost
时间 (单位: 秒)	4518.3	330.95	0.81	211.33	9.94

从上表能看出 CART 的运行时间最短，NaiveBayes 的时间最长 (因为使用了留一交叉方法来寻找最优参数)，而 Xgboost 的时间相比之下也较短，因此在实际操作中的实用性也很强。

5 总结

本文为了研究不同居民收入阶层的共性特征，选取了从美国人口普查局的数据库中提取的数据进行处理、建模和预测。数据集构成为 14 个自变量和 1 个我们关心的变量，在于处理中我们基于变量自身的含义分析以及变量间的相关性删除了两个自变量，最终划分成 80% 的训练集与 20% 的测试集。

本文在划分后的训练集上基于 NaiveBayes、KNN、CART、RandomForest 和 Xgboost 五种算法，通过留一交叉、历史先验等方法对各模型要求的参数进行了优化，建立了各自的模型并对测试集进行预测，整理出二分类下的混淆矩阵，并计算评价模型效果的各个指标值以及绘制评价模型的曲线图，再加上计算出的模型总的运行时间，对这五种模型进行比较。结果发现 Xgboost 的效果相对最好，RandomForest 的效果预期接近且相差不大，而 NaiveBayes 模型的效果最差，在实际问题操作中可能并不适合，且 Xgboost 的运行时间不长，考虑各种成本因素，实际操作中施行 Xgboost 方法进行处理可能是最佳选择。

参考文献

[1] 夏泽宽.从居民收入视角测度广东区域协调发展[J].中国国情国力,2019(05):67-71.

[2] 沈晨昱.XGBoost 原理及其应用[J].计算机产品与流通,2019(03):90.

[3] 叶倩怡,饶泓,姬名书.基于 Xgboost 的商业销售预测[J].南昌大学学报(理科版),2017,41(03):275-281.

[4] 沈倩倩,邵峰晶,孙仁诚.基于 XGBoost 的乳腺癌预测模型[J].青岛大学学报(自然科学版),2019,32(01):95-100.

[5] 杨鹏,曾朋,赵广振,吕培培.基于 Logistic 回归和 XGBoost 的钓鱼网站检测方法[J].东南大学学报(自然科学版),2019,49(02):207-212.

[6] 余芳东.世界银行关于国家收入分类方法及问题探讨[J].中国统计,2016(06):32-35.

[7] 李秉强. 我国居民收入增长及其影响因素研究[D].华中科技大学,2007.

[8] Lawrence H. Goulder,Marc A.C. Hafstead,GyuRim Kim,Xianling Long. Impacts of a carbon tax across US household income groups: What are the equity-efficiency trade-offs?[J]. Journal of Public Economics,2019,175.

[9] Chen T , Guestrin C . XGBoost: A Scalable Tree Boosting System[J]. 2016.

[10] Lawrence H. Goulder,Marc A.C. Hafstead,GyuRim Kim,Xianling Long. Impacts of a carbon tax across US household income groups: What are the equity-efficiency trade-offs?[J]. Journal of Public Economics,2019,175.

[11] Kohavi R . Scaling Up the Accuracy of Naive-Bayes Classifiers: a Decision-Tree Hybrid[J]. Proc. of KDD-96, 1996.

[12] Rosset S . Model Selection via the AUC[C]// International Conference on Machine Learning. ACM, 2004.