数据仓库大作业二

一、银行直销数据分类

1.作业要求

根据银行客户的属性判断客户是否会订阅某项定期存款业务，不限定分类算法，要求至少实现两种算法，并且对于算法精度进行分析比较

2.分类评价方法

实际 预测	Y	N
Y	a	b
N	c	d

如图所示，分别统计四种分类结果，并做如下整理：

$$
\begin{aligned}
Accuracy &= \frac{a+b}{a+b+c+d} \\
Precision &= \frac{a}{a+c} \\
Recall &= \frac{a}{a+b}
\end{aligned}
$$

考虑到本次分类的情景，为了尽可能多的找到购买服务的客户，应当在保证准确率的同时，尽可能增加真阳率，也就是增加类别为真的预测个数，甚至可以牺牲部分准确率来提高真阳率

3.实验环境

操作系统：Windows10 专业版 64位

处理器：Intel(R) Core(TM) i7-8700 @3.2GHZ

内存：16G

编程语言：Python 3.7

4.决策树分类

1）算法描述

对于所有的数值型属性，以平均值为轴，将所有的用例分为两类；对于所有的非数值型属性，每一个属性值分为一类。

以上述分类标准，为当前样例集合，计算当前属性集合中每一个属性的信息增益，找到信息增益最高的属性。如果是数值型属性，则按照大于平均值和小于等于平均值分出两棵子树；如果是非数值型数据，则为每一个属性值分出一棵子树。子树包含原样例集合中，符合分类标准的所有样例，属性集合中删除当前分类属性。递归为子树进行分类。

如果属性集合中属性数量为零，或者样例集合中样例数量为零，或者正负样例的比例达到了给定阈值，则不再对当前子树进行分类。

2）属性选择

选择了全部属性进行分类依据。

选择age， duration， campaign， pdays， previous， emp.var.rate， cons.price.idx， cons.conf.idx， euribor3m， nr.employed 作为数值型属性处理；

选择job， marital， education， default， housing， loan， contact， month， day_of_week， poutcome 作为非数值型属性处理

3）数据集选择和复杂度分析

设第i棵子树含有n_i个样例和k_i个属性，那么在找具有最大信息增益的属性时，必须遍历所有属性和所有样例，复杂度$n_i k_i$，在决策树的每一层中，所有子树的总体时间复杂度为$\sum n_i k_i = |K|n$，|K|代表当前子树的属性集合中的属性数目，因此算法总的时间复杂度是$O(nk^2)$。

复杂度较低，因此可以选择完整数据集。

4）正负例平衡方法

一般的决策树算法中，只有当子树中所有的样例都属于同一类时，也就是某一类的比例达到100%时，才停止子树的划分，但是在本次情景中，正样例非常少，因此当子树中样例集合的正例比例达没有达到100%时，在某一中间阈值，也可以停止子树的划分，并且将落到该子树的样例划分为正例。

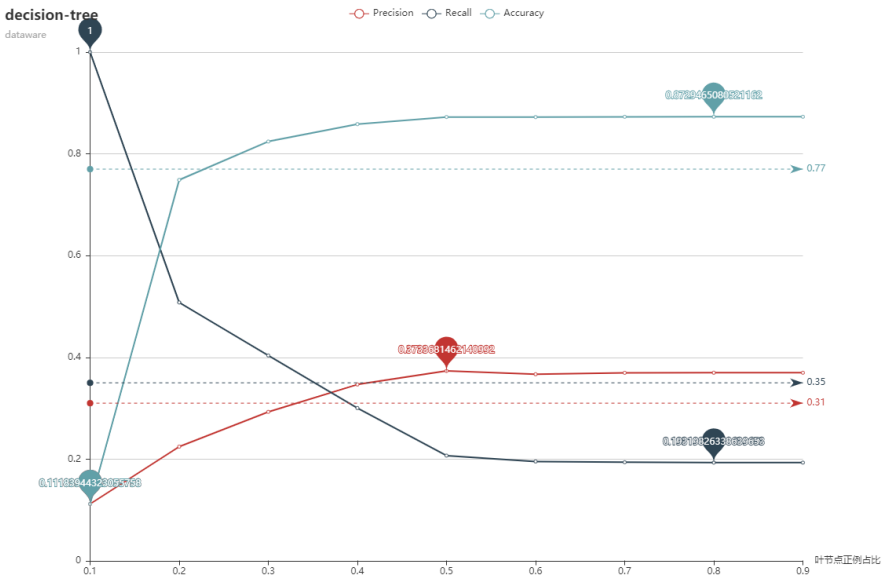
5）分类结果

对于大样本bank-additional-full，在正例占比0.6时停止子树分裂，准确度达到0.7440317229100913

对于小样本bank-additional，当正例的比例达到P时，停止子树的分类，分类的真阳率和准确度如下：

正例占比	真阳率$(a/(a+b))$	正确率
0.9	0.11940298507462686	0.866504854368932
0.6	0.11940298507462686	0.866504854368932
0.4	0.208955223880597	0.8519417475728155
0.3	0.26865671641791045	0.8252427184466019
0.2	0.3805970149253731	0.7459546925566343

对于大样本，当正例的比例达到P时，停止子树的分类，对应的Precision，Recall，Accuracy如下



5.朴素贝叶斯分类

1）算法描述

用A_{ij}表示A第i个属性的第j中值，为所有属性的所有值计算$P(A_{ij}|Y)P(A_{ij}|N)$

$$\vec{a} = [a_1, a_2, a_3, \ldots, a_{20}]$$
$$P(Y|\vec{a}) = \frac{P(a_1|Y)P(a_2|Y)\ldots P(a_{20}|Y)P(Y)}{P(a_1)P(a_2)\ldots P(a_{20})}$$
$$P(N|\vec{a}) = \frac{P(a_1|N)P(a_2|N)\ldots P(a_{20}|N)P(N)}{P(a_1)P(a_2)\ldots P(a_{20})}$$

如果$$P(Y|A_{ij}) >= P(N|A_{ij})$$，则将样例a分类为Y，否则分类为N

2）属性选择

选择了全部属性进行分类依据。

选择age，duration，campaign，pdays，previous，emp.var.rate，cons.price.idx，cons.conf.idx，euribor3m，nr.employed 作为数值型属性处理；

选择job， marital， education， default， housing， loan， contact， month， day_of_week， poutcome 作为非数值型属性处理

3）数据集选择和复杂度分析

在计算$P(A_{ij}|Y)$和 $P(A_{ij}|Y)$时，需要遍历所有n个样例的全部k个属性，时间复杂度$O(nk)$，在分类时，需要两次遍历所有样例的所有属性计算其为Y的概率和为N的概率，时间负责度为$O(2nk)$，因此总的时间负责度是$\Theta(nk)$

4）正负例平衡方法

在朴素贝叶斯分类中，正负样例的绝对数量并不影响分类的结果，因为计算$P(A_{ij}|Y)$和 $P(A_{ij}|Y)$时，已经包含了样本数量这个因素，样本数量少可能会导致容错性降低，不会导致系统性错误。

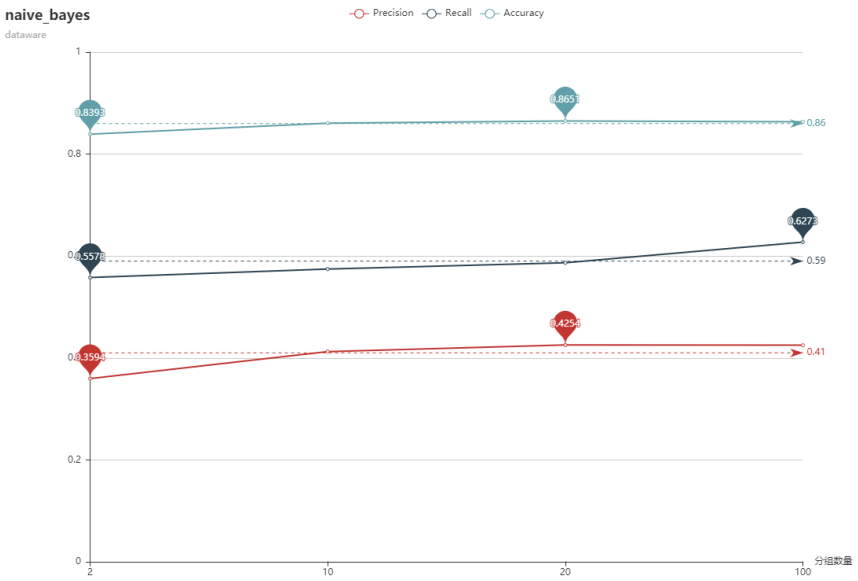
但是考虑到分类的实际情景，为了增大真阳率，适当的牺牲假阳率是可以的。因此可以通过改变分类条件，即将分类条件$$P(A_{ij}|Y) >= P(A_{ij}|N) * P,P\in [0,1)$$时将样例a分类为Y，来增大真阳率。

5）分类结果

对于小样本集bank-additional，分类正确可以达到0.8923948220064726

对于大样本集bank-additional-full，分类准确度可以达到0.6598038263571915

对于大样本，改变数值型属性的分组策略，会影响到最终的分类结果。下图是在等宽分组策略下，对应的Precision，Recall，Accuracy



6.K-NN分类

1）算法描述

选择和样例a距离最近的k个样例，如果正例占多数，则将a分类为Y，否则将a分类为N。

将样例的所有属性划分为数值型属性和非数值型属性，A_i^{max}表示第i个数值型属性的最大值，则样例a和b的在第i个属性生的距离定义为

$$dis = \begin{cases} (\frac{a_i - b_i}{A_i^{max}})^2 & 若\ A_i是数值型属性 \\ a_i == b_i ? 1 : 0 & 若\ A_i是非数值型属性 \end{cases}$$

2）属性选择

选择了全部属性进行分类依据。

选择age， duration， campaign， pdays， previous， emp.var.rate， cons.price.idx， cons.conf.idx， euribor3m， nr.employed 作为数值型属性处理；

选择job， marital， education， default， housing， loan， contact， month， day_of_week， poutcome 作为非数值型属性处理

3）数据集选择和复杂度分析

对于测试集中的每一个样例，都需要计算其与训练集中每一个样例的距离，复杂度为$\Theta(Nk)$，因此为测试集中的每一个样例分类的复杂度是$\Theta(nNk)$，这个复杂度是很高的，因此K-NN分类算法在这一情景下只适用于小样本集

4）正负例平衡方法

在K-NN分类中，k值的选择是十分重要的，特别是在正负样例数量相差较大的情景下，因为如果k选择过大，那么k个最近邻样本中一定是负样例占多数。如果是为了增大真阳率而牺牲假阳率，那么还可改变分类条件，例如k近邻样本中正例占比达到40%就判断为正例。

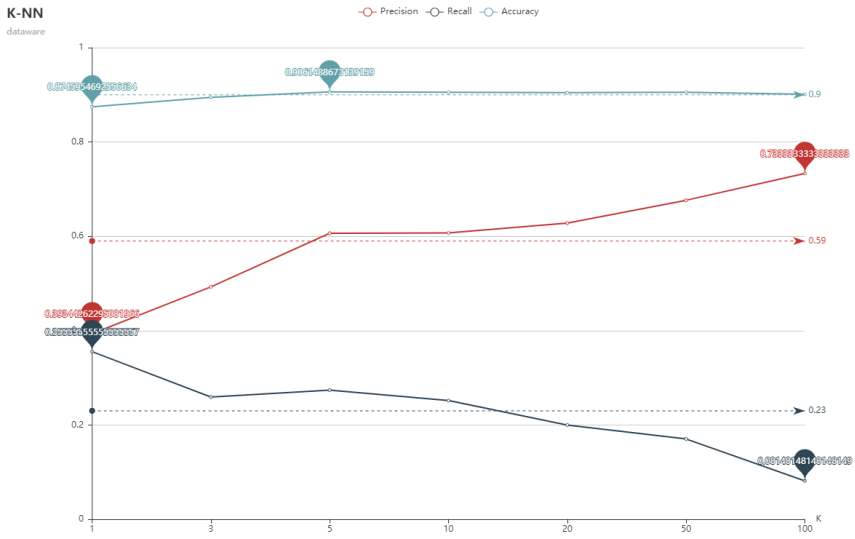
5）分类结果

k=10，只要有正例就判断为Y时，准确度为0.8358333333333333，真阳率(a/(a+b))为0.6259541984732825

k=100，正例占比达到10%就判断为Y时，准确度为0.8333333333333334，真阳率(a/(a+b))为0.7480916030534351

k=100，正例占比达到20%就判断为Y时，准确度为0.8933333333333333，真阳率(a/(a+b))为0.3969465648854962

对于大样本，在不同的K值时，K-NN分类对应的Precision，Recall，Accuracy如下



7.逻辑回归分类

1）算法描述

为每一个数值型属性设置一个权重值weight

$$result = sigmoid(\overrightarrow{cur}.transpose * \overrightarrow{weights})$$

result为1则分为Y，否则分为N

2）属性选择

逻辑回归方法仅适用于数值型数据，因此只使用了数值型属性，即age，duration，campaign，pdays，previous，emp.var.rate，cons.price.idx，cons.conf.idx，euribor3m，nr.employed 作为数值型属性处理。

3）数据集选择和复杂度分析

权重值的每一次更新迭代，都需要用当前的权重值计算每一个训练样本的类别，这一过程的复杂度是O(nk)，但是这一过程使用矩阵乘法可以快速完成。权重值的更新次数c是手动设置的，因此算法的总的时间复杂度是O(cnk)，可以用于大样本集

4）正负例平衡方法

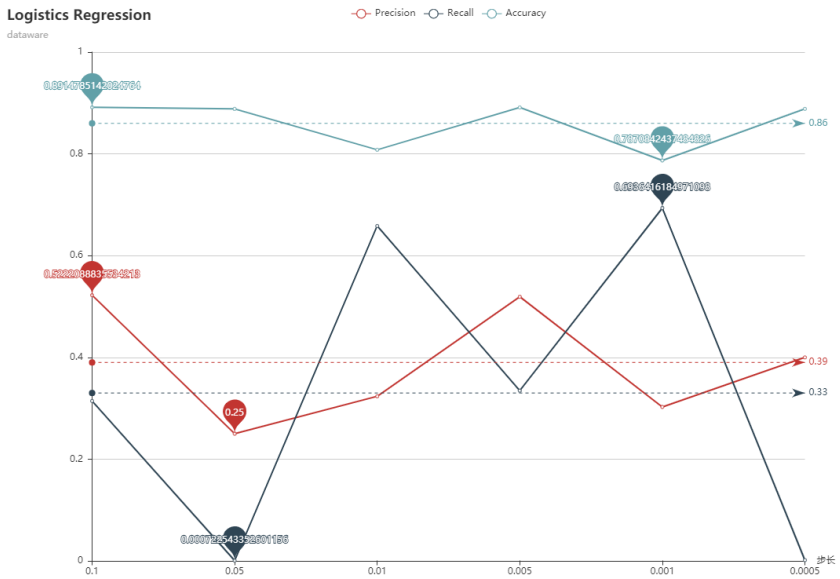
逻辑回归算法的根本思想是找到正负样例之间的分界，因此几乎不会受到样本数量的影响，同时也很难照顾样本数量因素，因此没有使用特别的方法平衡正负样例。

5）分类结果

在迭代步长为0.001，迭代次数为5000次时，对大样本的分类准确度是0.7651743226182384

在迭代步长为0.001，迭代次数为5000次时，对小样本的分类准确度是0.9087378640776699

对于大样本，不同的步长会对逻辑回归分类的结果造成影响，在不同步长下，对应的Precision，Recall，Accuracy如下



8.四种分类方法比较

1.适用范围

决策树算法，逻辑回归算法和朴素贝叶斯算法执行速度较快，适用于大数据集，而K-NN算法执行速度较慢，适用于小数据集。

决策树算法，朴素贝叶斯算法和K-NN算法都可以支持非数值型属性，但是K-NN算法不支持非数值型属性，但是必要时可以将非数值型属性转化为数值型属性。

2.准确度

四种分类方法的准确度比较

分类方法	准确度（大）	准确度（小）	真阳率（大）	真阳率（小）
决策树	0.8729465080521162	0.866504854368932	0.3812154696132597	0.25396825396825395
朴素贝叶斯	0.8609694909767743	0.88915857605178	0.3443868099296036	0.4919786096256685
$K-NN$	–	0.9053398058252428	–	0.3565217391304348
逻辑回归	0.8912357368293274	0.9087378640776699	0.3333333333333333	0.6119402985074627

3.真阳率

决策树算法、K-NN算法对正负样本的比例敏感，在正例较少的情况下，容易造成真阳率低，但是这两种算法都能够通过调整分类细节的方式来降低正负样本比例的影响。

朴素贝叶斯算法受正负样本比例影响较小，但是也能够通过调整分类细节达到牺牲假阳率，提高真阳率的效果。

逻辑回归算法受正负样本比例影响较小，且很难通过调整提高真阳率。

4.可解释性

四种分类方法均有较强的解释性。

决策树算法根据每一个非叶节点的判断标准可以知道如何一步一步进行分类，这些在每一个维度的分类标准合起来就是对整个样本的分类标准；

K-NN算法找到的是与测试样本最为相近的训练样本，根据与其最为相似的样本的类别来决定当前样本的类别；

逻辑回归算法中每一个属性的权重值与属性值范围综合考虑，代表了某一个属性的重要程度和对分类结果的贡献；

朴素贝叶斯算法直接给出了每一个属性对最终分类结果的贡献。