**刘昕**

**苏莉娅**

**蒋浩丰**

**昝文**

**2016年12月3日**



**针对乳腺癌患者数据，建立模型，围绕各个病程的特殊症候给予分类分析**。

数据挖掘大作业报告

# 说明文档

[1. 项目概述及人员： 2](#_Toc15789)

[1.1项目概述 2](#_Toc31131)

[1.2 人员 2](#_Toc22722)

[2. 需求分析 2](#_Toc13032)

[3. 系统设计 3](#_Toc27732)

[3.1 总体设计 3](#_Toc7275)

[3.2 系统功能模块 4](#_Toc6979)

[4.代码框架及函数设计 11](#_Toc17618)

[4.1.实现环境： 11](#_Toc7058)

[4.2.主要函数设计 11](#_Toc3062)

[5.实验 14](#_Toc20498)

[5.1.实验环境： 14](#_Toc10841)

[5.2.实验步骤： 14](#_Toc13410)

[5.3实验结果分析： 24](#_Toc26013)

[6.实验总结 25](#_Toc22041)

1. 项目概述及人员：

## 1.1项目概述

针对乳腺癌患者，可以依据中医阶段治疗法进行治疗，在辨症的基础上围绕各个病程的特殊症候先症而治型。依据医学指南，将乳腺癌辨证统一化，分为六种症型。即患者在围手术期、围化疗期、围放疗期和内分泌治疗期，基本都会出现特定的临床症状。可以通过一定的规则，挖掘出各中医症素与乳腺癌TNM分期之间的关系。探索不同分期阶段的乳腺癌患者的中医症素分布规律，以及阶段病变发展（71，72，73），指导乳腺癌的中医临床治疗。

（1）对数据进行预处理；

（2）利用数据挖掘的算法对数据进行分析，得到中医症素与乳腺癌TNM 分期之间的关系。根据实际的业务分析建议最小支持度设定为：6%，最小置信度为：75%；

（3）对所构建的模型进行分析，得出你的结论。

## 1.2 人员

|  |  |  |
| --- | --- | --- |
| 姓名 | 学号 | 分工 |
| 刘昕 | 2016E8018661172 | 贝叶斯算法，文档 |
| 蒋浩丰 | 201628018629015 | 神经网络，频繁集 |
| 苏莉娅 | 201628018629065 | 决策树，框架设计，预处理 |
| 昝文 | 201628018629015 | k近邻，图形界面 |
| 李博伟 | （协助未选课） | svm |

# 2. 需求分析

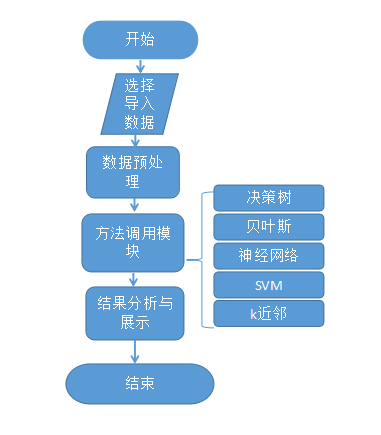
（1）功能需求：利用数据挖掘的算法对数据进行分析，得到中医症素与乳腺癌TNM 分期之间的关系。根据实际的业务分析建议最小支持度设定为：6%，最小置信度为：75%；对所构建的模型进行分析，得出你的结论。

（2）性能需求：满足多算法优势比较。满足正确率在90%以上，错误率低于10%，运算速度可接受。

采用图形化界面，有统一框架，可以集中调用算法进行输出。

# 3. 系统设计

## 3.1 总体设计



首先分析数据，获取数据的整体分布以及数据质量情况。

针对问题，归类为频繁集查找问题，也可考虑使用分类来解决。

通过离散化处理后使用频繁集发现算法-Apriori，发现数据项之间的关联性。对于连续值的频繁集查找需要考虑数据的维数扩展，以及合理的离散方法，设置为老师提供的最小支持度和最小自信率，发现获得了多为冗余的规则，甚至矛盾的规则，这些数据的可解释性非常差。考虑到问题的求解，我们变换思路，使用了分类的思路解决问题。

为了充分体验不同分类算法的特性，以及了解针对数据不同算法的效果情况，将学习的五种分类算法都做一次尝试，在初步尝试的基础上分析结果，进一步优化系统的正确率。

## 3.2 系统功能模块

### 3.2.1 预处理

在预处理阶段，我们采用箱均值的方法对数据进行光滑或者直接去除空值的方法去噪。经过比较,去除空值的方法准确率较高，故最终采用了去空值的方案去去除噪声。

### 3.2.2 频繁集发现算法：

采用Apriori，针对连续数据进行改进。

###### 原理：

Apriori算法的两个输入参数分别是最小支持度和数据集。该算法首先生成所有单个物品的项集列表，遍历之后去掉不满足最小支持度要求的项集；接下来对剩下的集合进行组合生成包含两个元素的项集，去掉不满足最小支持度的项集；重复该过程直到去掉所有不满足最小支持度的项集。

###### 实现：

输入加上输出一共用十位标志位表示，所以频繁集所有模式穷举可以使用0-1023表示（这1024位数字如果使用二进制表示，从低位到高位，如果该位为1，则表示十位标志位对应的位为1）设定一个固定频数值N，（即如果模式出现的次数>=N，说明该模式是频繁的）

第一步，统计0-1023每个数字所代表的模式存在的数量

举个例子，如67，其二进制表示为0001000111，其中第一、二、三、七位为1，那么如果十位标志位中满足第一、二、三、七位为1，其余位为0，则称满足条件，则67对应的计数+1

第二步，初始化初始数据项集C1

即将数字的二进制表示中有仅有两个位为1或一个位为1的数字放入其中

第三步，筛选出C1中对应的频数大于N中的项生成频繁项集L1

第四步，对L1中的项进行或运算，生成一个新的数据项集C2,重复第三步、第四步运算重复8次

注：进行或运算是要判定两个项的二进制表示只有两个位不一样并且该两个位置1在分别分布在两个数字上，其实现方式是让两个数相减得到的结果存在于一个列表中。例：305（0100110001）和297（0100101001）这两个数字只有第四、五为不一样，且置一位分别分布在两个数字上，则这两个数字可以进行或运算。

第五步：收集上述生成的频繁项集，对其进行过滤，最终生成最大频繁项集

过滤规则是：如果两个频繁集项对应的低六位相同，但高四为不同的话，说明同样的模式对应了两个输出，出现结果歧义，就将两个频繁项删除；如果两个频繁集项A,B，A的所有置一位，B全部置一，而B的置一位，A未全部置一，说明A不是更大的频集，就将A删除；

### 3.2.3 分类算法：

我们采用了决策树、朴素贝叶斯、SVM、K-近邻、神经网络的方法对训练样本进行分类学习。

#### 决策树：

###### 原理：

决策树是一个预测模型，代表的是对象属性与对象值之间的一种映射关系。树中每个节点表示某个对象，而每个分叉路径则代表的某个可能的属性值，而每个叶结点则对应从根节点到该叶节点所经历的路径所表示的对象的值。决策树一般自上而下生成的。分割的目的一致：对目标类尝试进行最佳的分割。

首先利用训练数据集来构造一棵决策树，树建立起来就可为未知样本产生一个分类。决策树模型有很多的优点、便于使用，高效、决策树的大小独立于数据库、、可以对有许多属性的数据集构造决策树。但决策树模型也有过拟合等问题的缺点。

###### 实现：

本文采用的是基于ID3决策树的实现方式针对连续数据进行的改进版本。D3算法的核心思想是在决策树的每一个非叶子结点划分之前，先计算每一个属性所带来的信息增益，选择最大信息增益的属性来划分。因为信息增益越大，区分样本的能力就越强，越具有代表性。该算法采用自顶向下的贪婪搜索遍历可能的决策空间。针对连续数据，需要针对每个属性的所有值排序后，在每个数据中间进行分割，求出最优的分割点，进而判断选择哪个属性来分割。

#### 朴素贝叶斯：

###### 原理：

贝叶斯分类器的分类原理是通过某对象的先验概率，利用贝叶斯公式计算出其后验概率，即该对象属于某一类的概率，选择具有最大后验概率的类作为该对象所属的类。

朴素贝叶斯模型拥有稳定的分类效率,所需估计的参数很少，对缺失数据不太敏感，算法也比较简单。因为NBC模型假设属性之间相互独立，这个假设在实际应用中往往是不成立的，这给NBC模型的正确分类带来了一定影响。在属性相关性较小时，NBC模型的性能最为良好。

###### 实现：

朴素贝叶斯的三个常用模型：高斯、多项式、伯努利。实现,本次的实验数据为连续值，适合采用高斯分布的方式进行计算。

本次采用这个公式为基础实现朴素贝叶斯的高斯模型。其中Xi代表着测试集中的每一项,

实现该函数时遇到了以下问题：

1. 在数据集读入时遇到存储格式的问题,本算法采用numpy的genfromtxt方式进行数据集的读取。但读取后文件格式为ndarray,和python的list类型不同,ndarray的数组内不允许存在多种数据类型的元素。但属性和类别分开存储不便于后面的分类及预测。

所以在这里我们采用ndarray的tolist()功能,将数据从ndarray格式转换成list格式，再按顺序进行拼接。

# 读取数据集  
dataMat = genfromtxt(freadn, delimiter=**','**, usecols=(0, 1, 2, 3, 4, 5))  
dataMat = dataMat.tolist()  
# 病程阶段  
target = genfromtxt(freadn, delimiter=**','**, usecols=(6), dtype=str)  
target = target.tolist()

#list拼接  
count = 0

*while* count < len(dataMat):  
 dataMat[count].append(target[count])  
 count = count+1

2.在核心函数高斯分布实现时读入数据集时采用list类型进行存储，存储格式只有最末项类别处为字符串类型,其余均为浮点数。实现高斯分布的基本公式需要引入python的math库并计算样本的均值和标准差，而计算公式不能处理最末项为字符串的情况。

所以我们采用了笨方法,增加了一个比较条件，遇到’Hi’类型的字符串（i=1,2,3,4）时,均值函数返回’Hi’,标准差函数返回0,

# 均值

*def* mean(*numbers*):  
 *if* (*numbers*[0] == **'H1'**)*or*(*numbers*[0] == **'H2'**)*or*(*numbers*[0] == **'H3'**)*or*(*numbers*[0] == **'H4'**):  
 *return numbers*[0]

.......

# 标准方差  
*def* stdev(*numbers*):  
 *if* (*numbers*[0]== **'H1'**) *or* (*numbers*[0] == **'H2'**) *or* (*numbers*[0] == **'H3'**) *or* (*numbers*[0] == **'H4'**):  
 *return* 0

......

从而避开了计算公式不能处理字符串的情况

#### SVM：

###### 原理：

支持向量机(support vector machine)是一种分类算法，通过寻求结构化风险最小来提高学习机泛化能力，实现经验风险和置信范围的最小化，从而达到在统计样本量较少的情况下，亦能获得良好统计规律的目的。

SVM是一种二类分类模型，其基本模型定义为特征空间上的间隔最大的线性分类器，即支持向量机的学习策略便是间隔最大化，最终可转化为一个凸二次规划问题的求解.

###### 实现：

我们实现SVM的方法是调用libsvm的库。LIBSVM是台湾大学林智仁(LinChih-Jen)副教授等开发设计的一个简单、易于使用和快速有效的SVM模式识别与回归的软件包。由于libSVM程序小，运用灵活，输入参数少，并且是开源的，易于扩展，因此成为目前国内应用最多的SVM的库。

#### K-近邻：

###### 原理：

K最近邻(k-Nearest Neighbor，KNN)分类算法的思路是：

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。

KNN算法中，所选择的邻居都是已经正确分类的对象。该方法在定类决策上只依据最邻近的一个或者几个样本的类别来决定待分样本所属的类别。

KNN方法在类别决策时，只与极少量的相邻样本有关。由于KNN方法主要靠周围有限的邻近的样本，而不是靠判别类域的方法来确定所属类别，KNN方法更适合类域的交叉或重叠较多的待分样本集。

KNN算法的主要不足是，当样本不平衡时，如一个类的样本容量很大，而其他类样本容量很小时，有可能导致当输入一个新样本时，该样本的K个邻居中大容量类的样本占多数。KNN算法的另一个不足之处是计算量大。该算法比较适用于样本容量比较大的类域的自动分类，而那些样本容量较小的类域采用这种算法比较容易产生误分。

###### 实现：

设定参数k,并计算范式距离。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，遍历并比较训练元祖集并不断更新，确保集内为范式距离最小的K个元祖。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。

#### 神经网络：

###### 原理：

神经网络分为前向传输和逆向反馈，我们实现的是逆向反馈及BP神经网络。训练一个BP神经网络，实际上就是调整网络的权重和偏置这两个参数，BP神经网络的训练过程分两部分：

前向传输，逐层波浪式的传递输出值；

逆向反馈，反向逐层调整权重和偏置；

###### 实现：

根据下面的公式



由上面两个公式算出隐藏层节点进而算出输出层节点

根据下面的公式算出输出层节点的偏差值

IMG_256

中Ej表示第j个结点的误差值，Oj表示第j个结点的输出值，Tj记录输出值

根据下面的公式算出隐藏层节点的偏差值

IMG_256Wjk表示当前层的结点j到下一层的结点k的权重值，Ek表示下一层的结点k的误差率

再根据下式更新各个节点的权值和偏置

IMG_256

IMG_256

如此对每一个训练节点进行训练，不断更新各个节点的权值，重复N次。

### 3.2.4 测试

我们采用的是交叉验证的方法划分测试集与训练集进行分类器的训练进行测试。

交叉验证的基本思想是把在某种意义下将原始数据(dataset)进行分组,一部分做为训练集(train set),另一部分做为验证集(validation set or test set),首先用训练集对分类器进行训练,再利用验证集来测试训练得到的模型(model),以此来做为评价分类器的性能指标.

本次实验我们采用的是K折交叉验证，初始采样分割成K个子样本，一个单独的子样本被保留作为验证模型的数据，其他K-1个样本用来训练。交叉验证重复K次，每个子样本验证一次，平均K次的结果最终得到一个单一估测。

### 3.2.4 功能集成：

我们通过web的图形界面进行分类器的集成和结果展示.

**实现：**

我们引入了flask的库实现web界面的开发。Flask是一个使用 Python 编写的轻量级 Web 应用框架，易于上手。

# 4.代码框架及函数设计

## 4.1.实现环境：

代码实现语言：python

编码环境：windows

## 4.2.主要函数设计

### 4.2.1数据预处理:

**代码路径：**

LIBOWEI/preprocessing.py

**主要功能函数：**

1>fillNullWithMean# 用均值填补空值

2>deleteOutliers #计算统计异常值边界,并去掉边界值

### 4.2.2 频繁集挖掘：

###### 代码路径：

jianghaofeng/frequent\_pattern.py

###### 主要函数：

1>handle\_data(self, one\_index, two\_index, three\_index, four\_index, five\_index, six\_index) #读取输入数据，并对数据进行转换等预处理操作

2>pre\_init\_data(self)#对处理过得数据进行统计，生成一个模式与频数的映射表

3>handle\_frequent(self)#频繁集生成算法Aprior实现

4>predict(self, frequent\_set\_lst)#使用最终的频繁集预测数据

### 4.2.3分类算法

#### 决策树：ID3算法

**代码路径：**

suliya/ID3ForContinuousValue.py

**主要功能函数：**

1>calcShannonEnt(dataSet):#计算数据集中的香农熵，看数据的标签分布

2>chooseBestFeatureAndValueToSplit(dataSet,labels):#找到最大熵增益的特征及特征取值

3>createTree(dataSet,labels):#创建决策树，使用熵增益评判

#### SVM:

**代码路径：**

libowei/svm.py

**主要功能函数：**

1>get\_SV(self):获得支持向量

2>svm\_train(arg1, arg2=None, arg3=None):SVM训练函数

3>svm\_predict(y, x, m, options=""):SVM预测函数

#### K近邻法：

**代码路径：**

zanwen/knn.py

**主要功能函数：**

1> classify(inX, dataSet, labels, k, lp):分类函数

2> getLpDistances(lp, inX, dataSet):# 计算两个向量的lp范数

#### 朴素贝叶斯：

**代码路径:**

liuxin/NB.py

**主要功能函数:**

1>summarizeByClass(dataMat):#通过计数统计获得分类器

2>calculateClassProbabilities(summaries, inputVector):#通过高斯函数计算概率。朴素贝叶斯高斯模型的核心算法。

#### 神经网络：

**代码路径：**

handle\_data.py

bp\_network\_test.py

**主要功能函数：**

handle\_data():#对xls数据进行处理，获取前六列与第八列数据放入一个TXT中，如果相应的行有空值，则舍去该行值。

input\_data():#用于获取TXT中的数据，并将其对变量赋值，随机选取900条数据中的700条作为训练数据，200条数据作为预测数据。

train():#用于训练数据，最终得到神经网络各个节点对应的参数矩阵。

predict():#用于预测数据，最终得到一个预测成功的数据的个数。

### 4.2.3 展示界面：

##### 代码路径:

zanwen/gui.py

# 5.实验

## 5.1.实验环境：

操作系统：windows

编程语言： python

## 5.2.实验步骤：

### 1)数据处理

本文的数据中包含着一些空值，需要通过数据预处理的方式处理噪音。

### 2)特征选择：

按照题目要求，本文选择的特征为“肝气郁结证型系数”，“热毒蕴结证型系数”，“冲任失调证型系数”，“气血两虚证型系数”，“脾胃虚弱证型系数”，“肝肾阴虚证型系数”，并以TNM分期为目标进行分类学习。

### 3)调用分类方法

在对数据进行预处理去除噪音之后，划分训练集和测试集，分别调用KNN，朴素贝叶斯,决策树，神经网络,SVM方法训练分类器

### 4）运行并测试

对测试集运行分类器，检验分类结果。并通过交叉验证法测定正确率和覆盖率，检验分类器的效果。

### 5）展示：

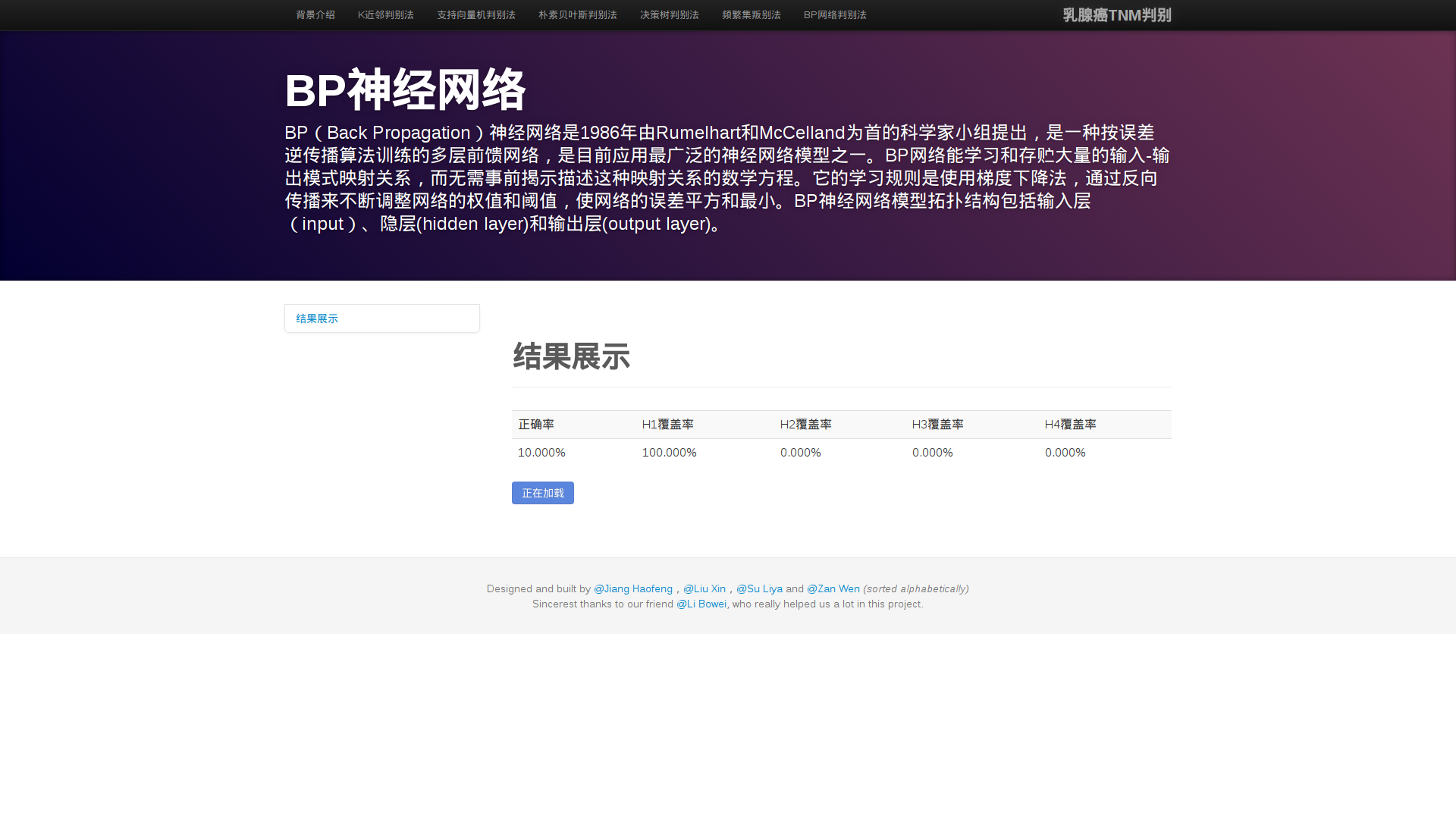
通过图形界面集中展示实验结果。

* 背景介绍：

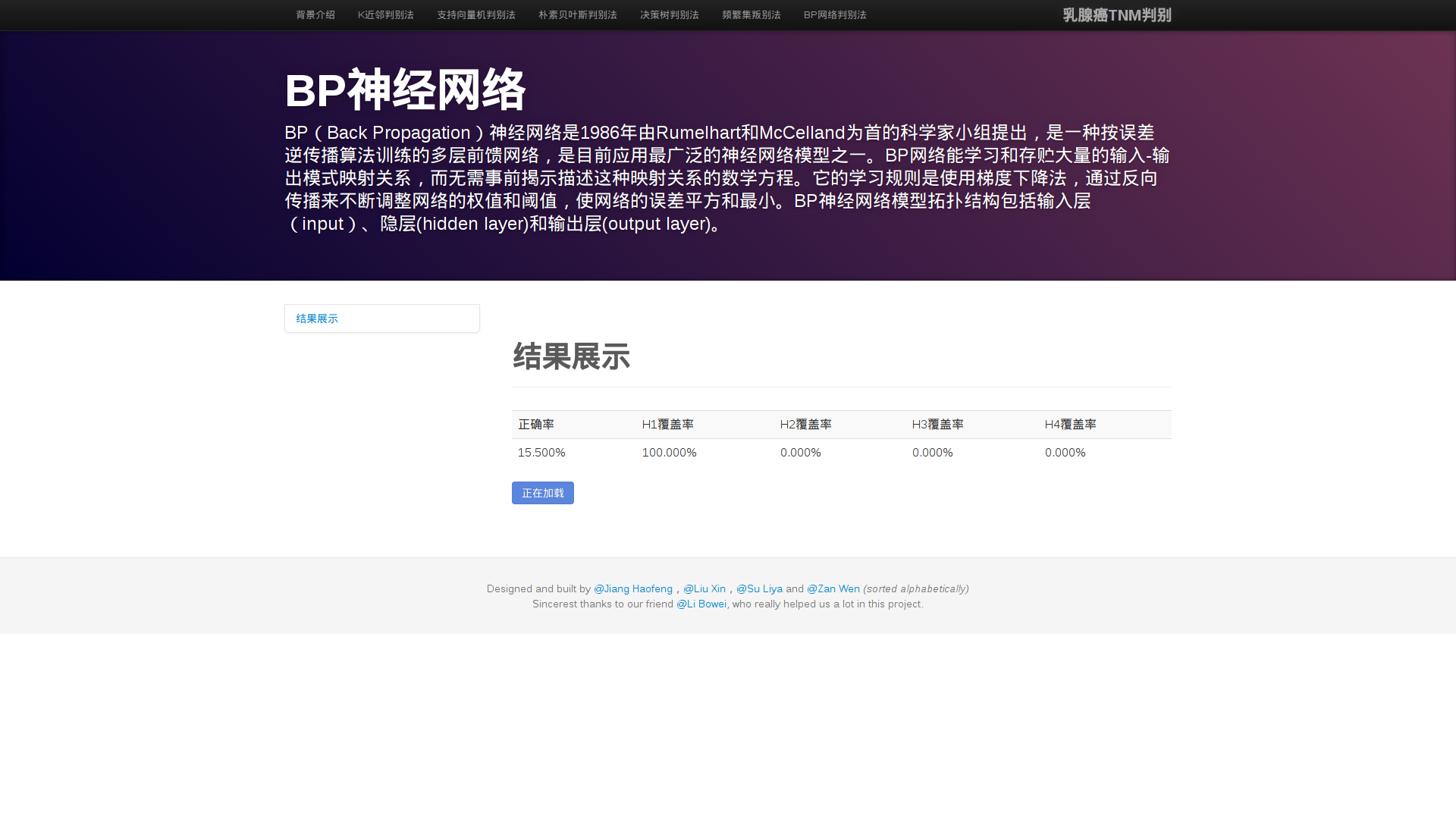


* BP网络

首次运行结果

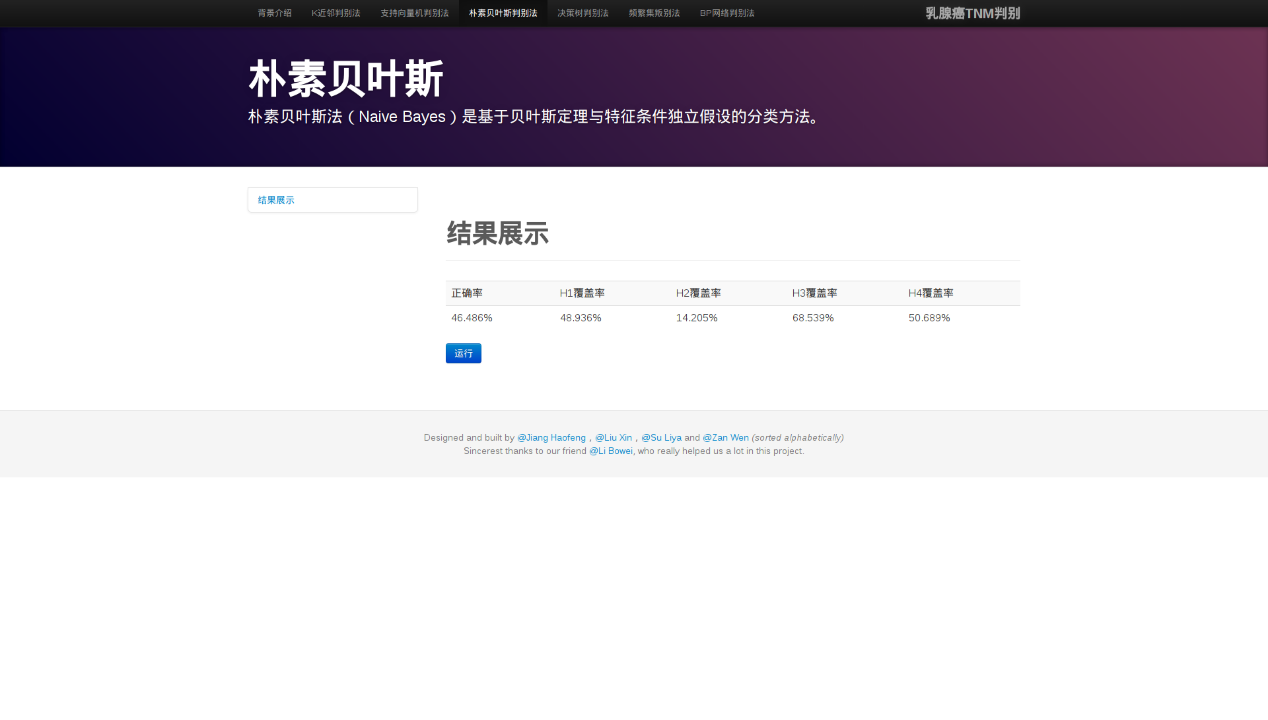


第二次运算结果

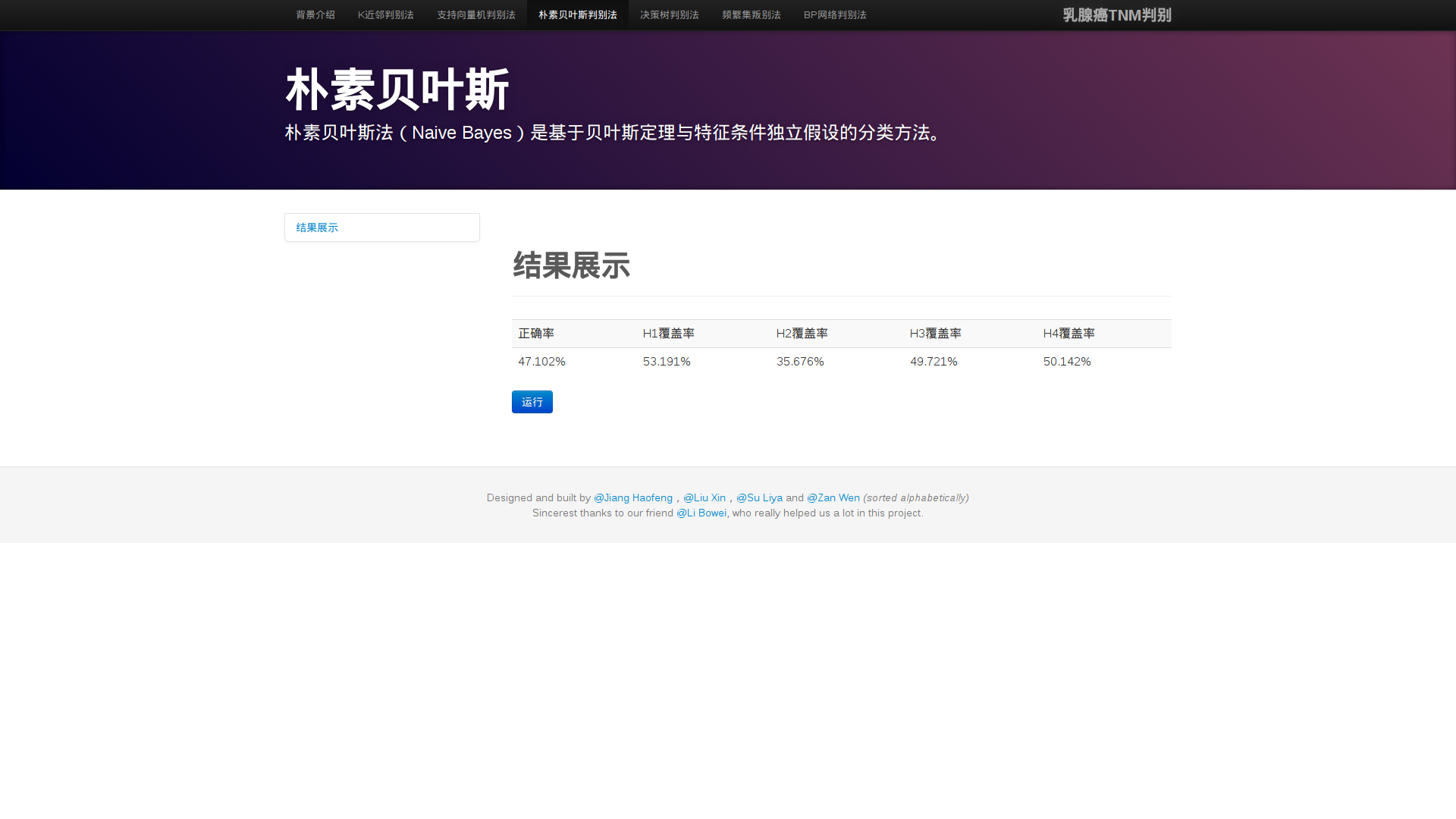


* 朴素贝叶斯

首次运行结果

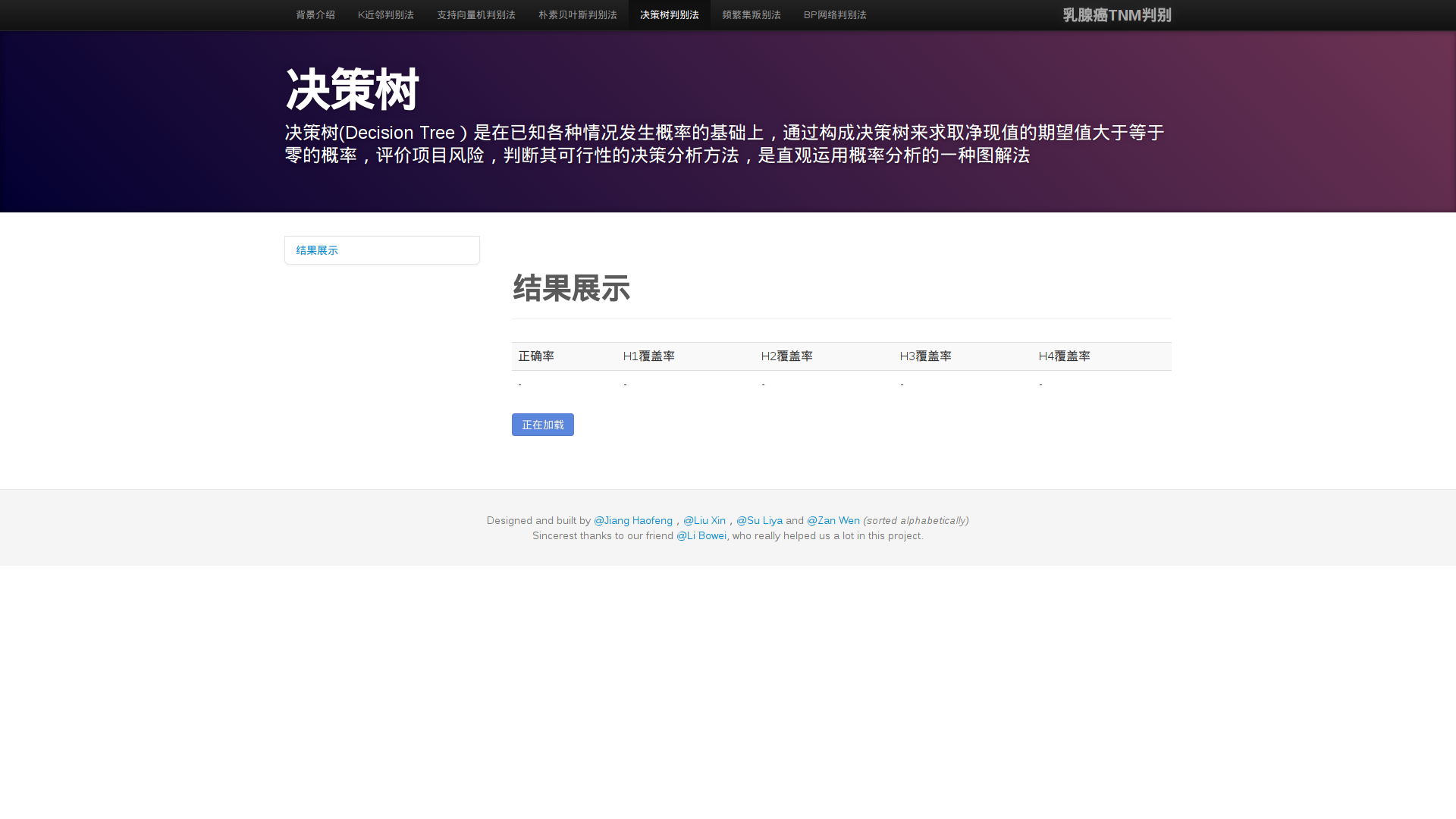


第二次运行结果

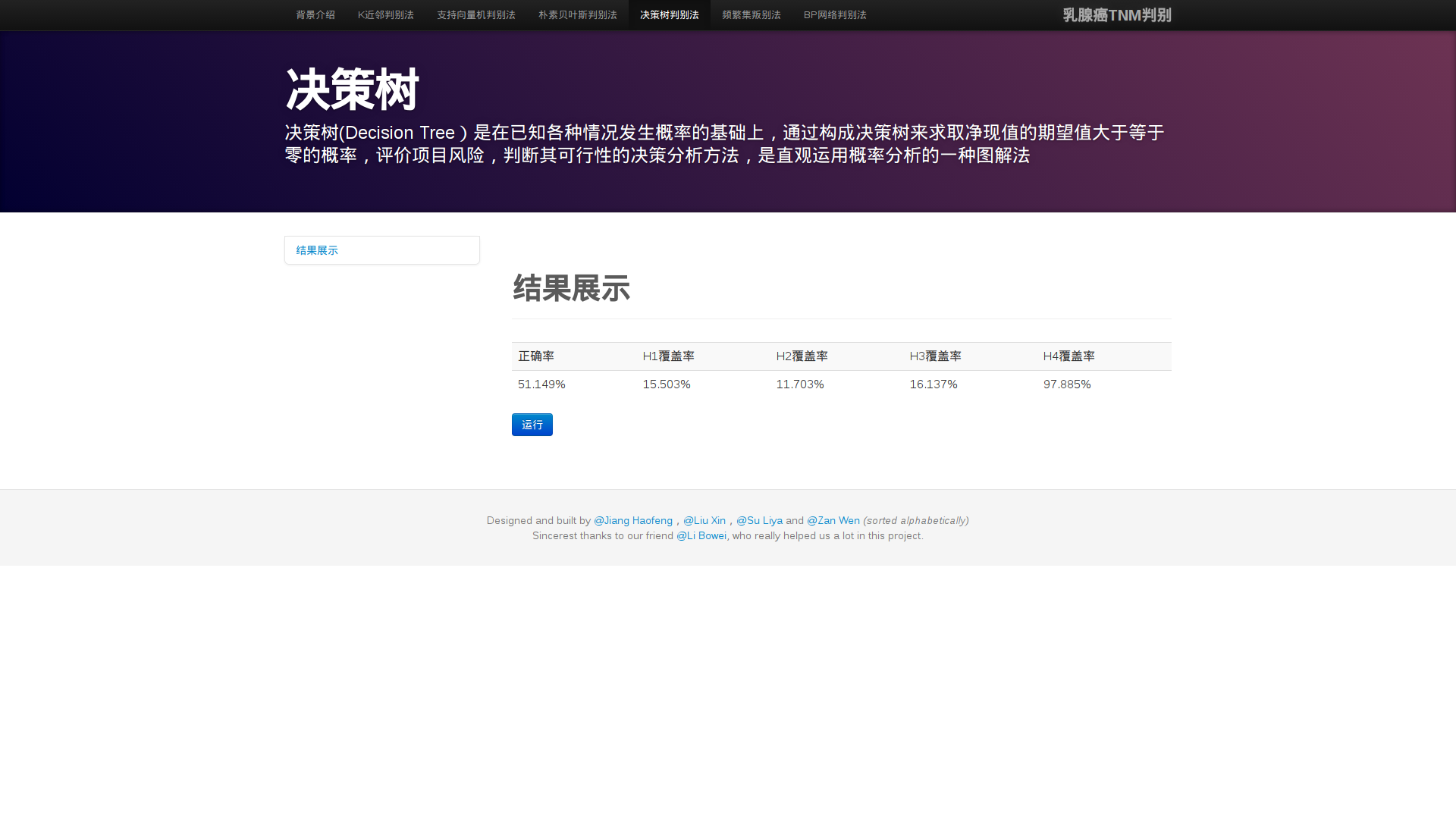


* 决策树

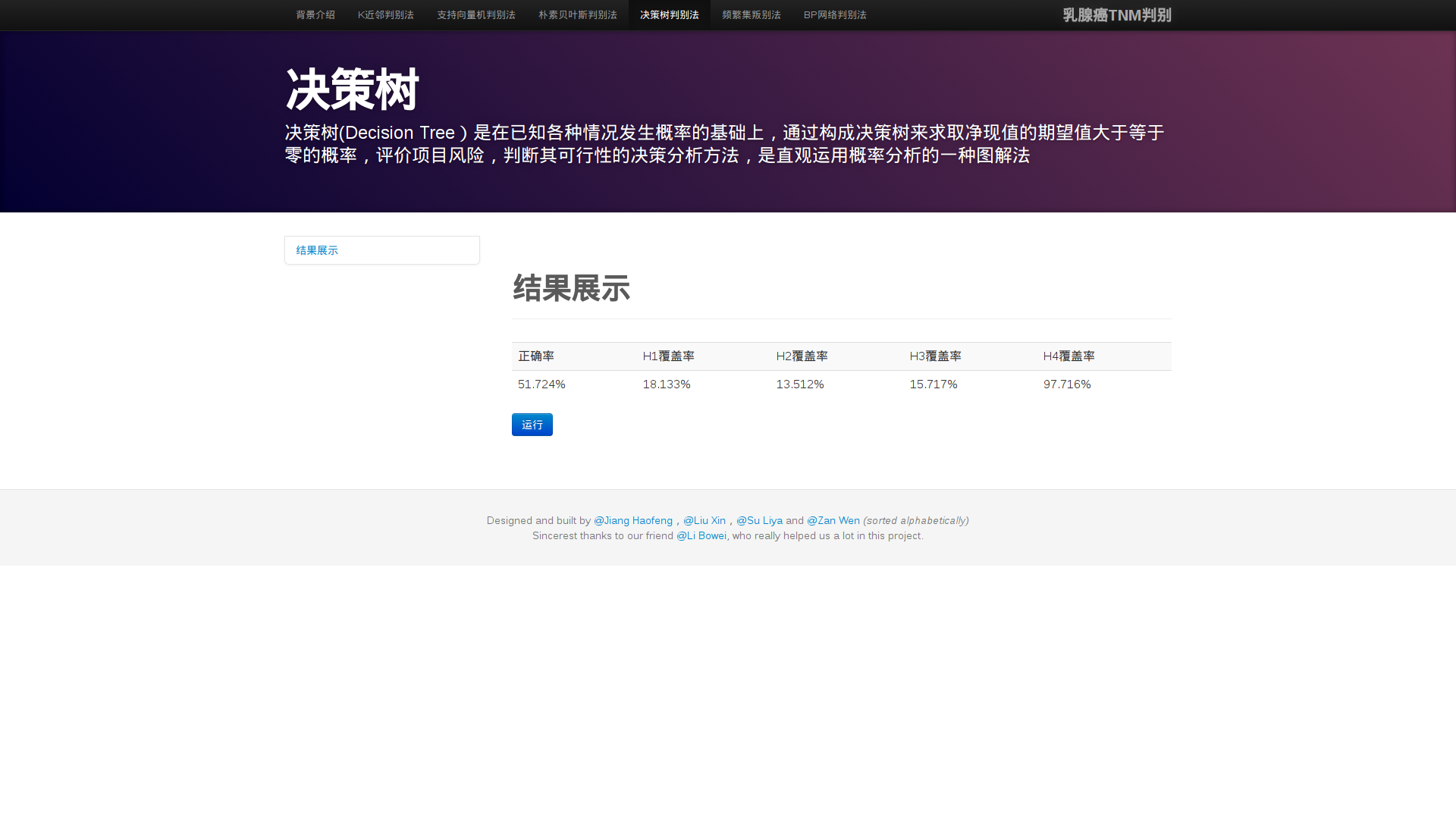
决策树生成较慢，点击运行会提示“正在加载”



得到结果



第二次运算结果



* K近邻

参数k为邻居个数，lp为距离范数，初始化都为1



设置k=2, lp=2



点击运行



修改参数k=10, lp=10，并运行



设置k=50, lp=50并运行

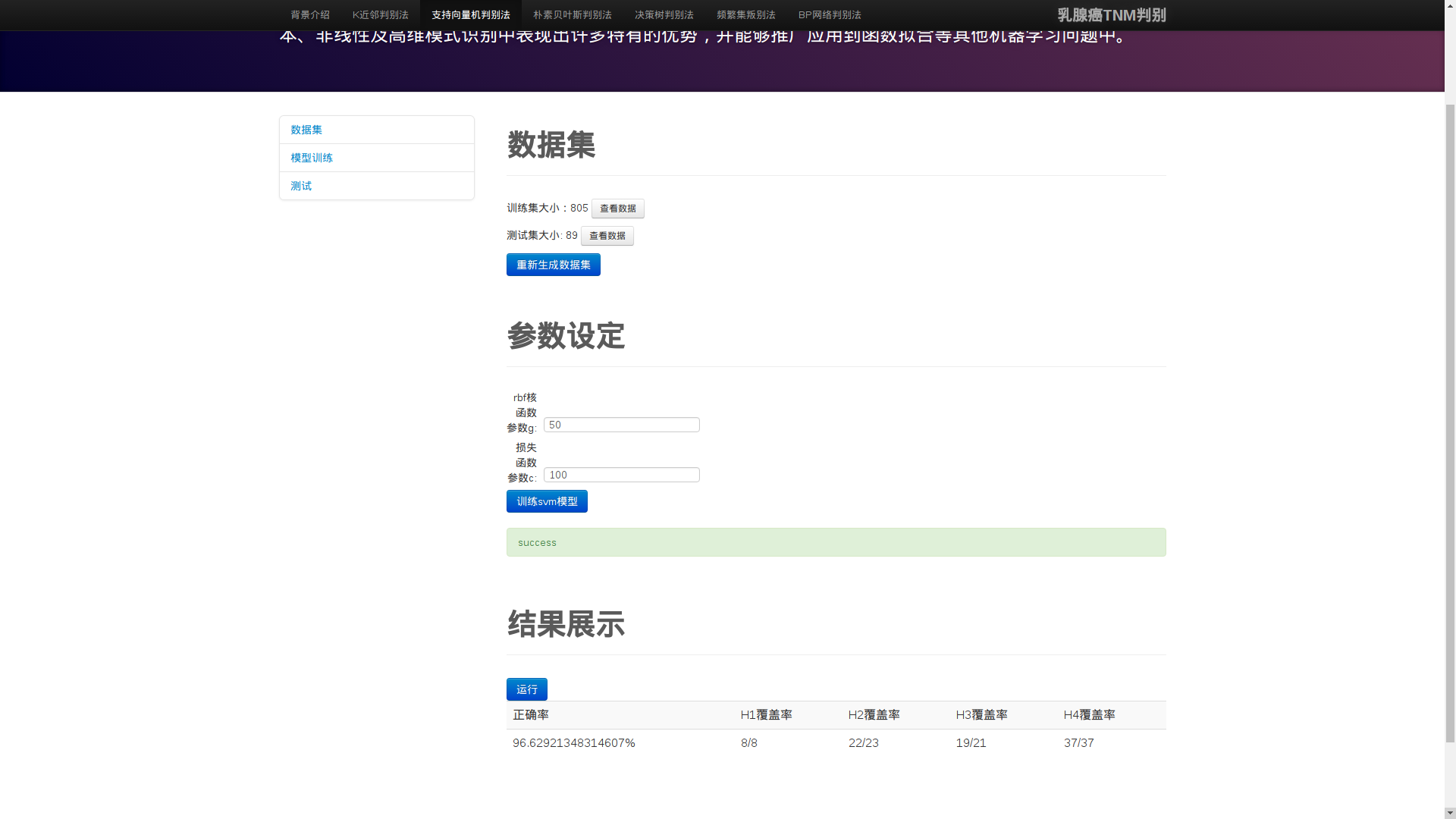


* 支持向量机

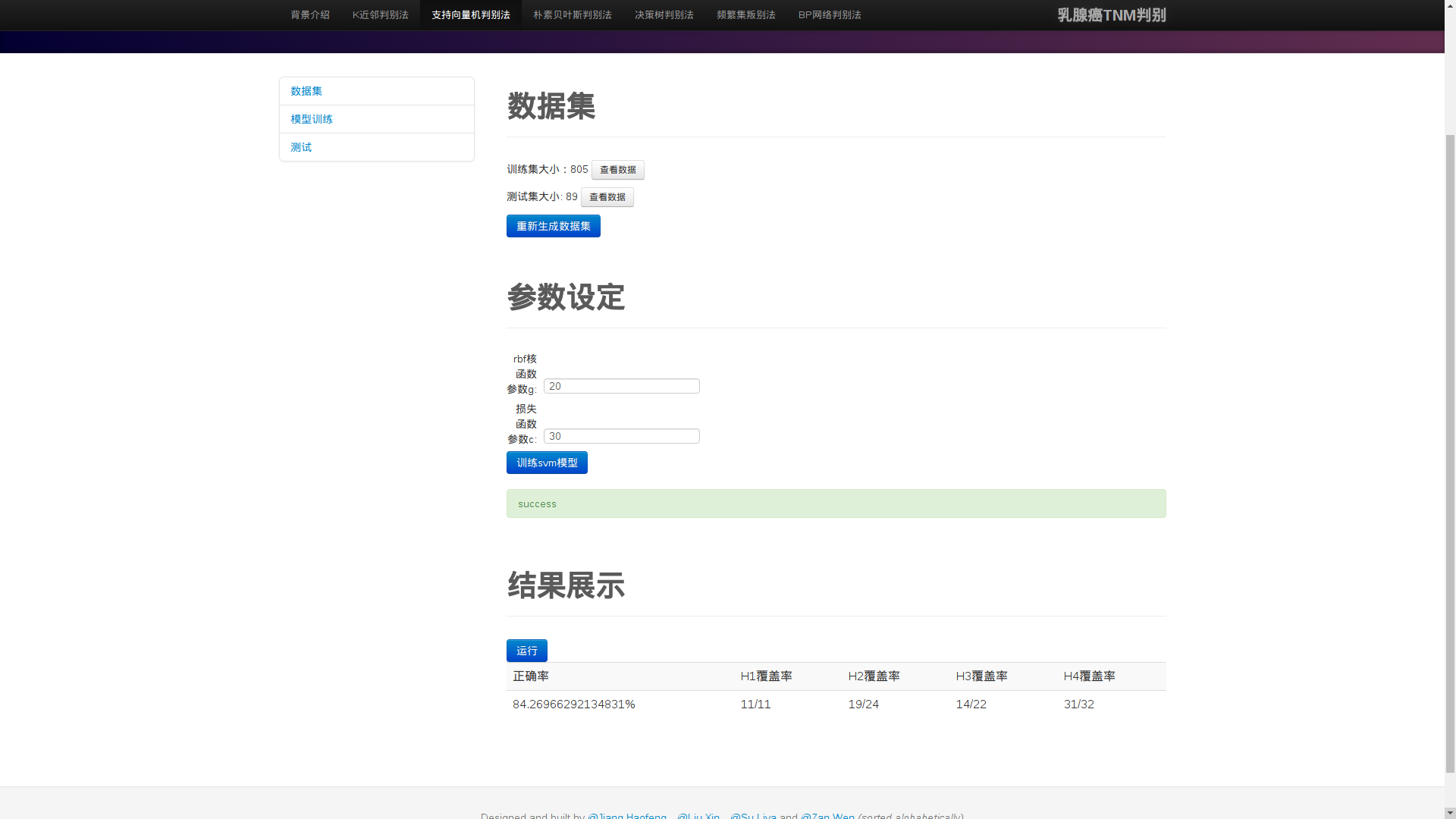
初始化参数: g=50, c=100



训练svm模型并运行

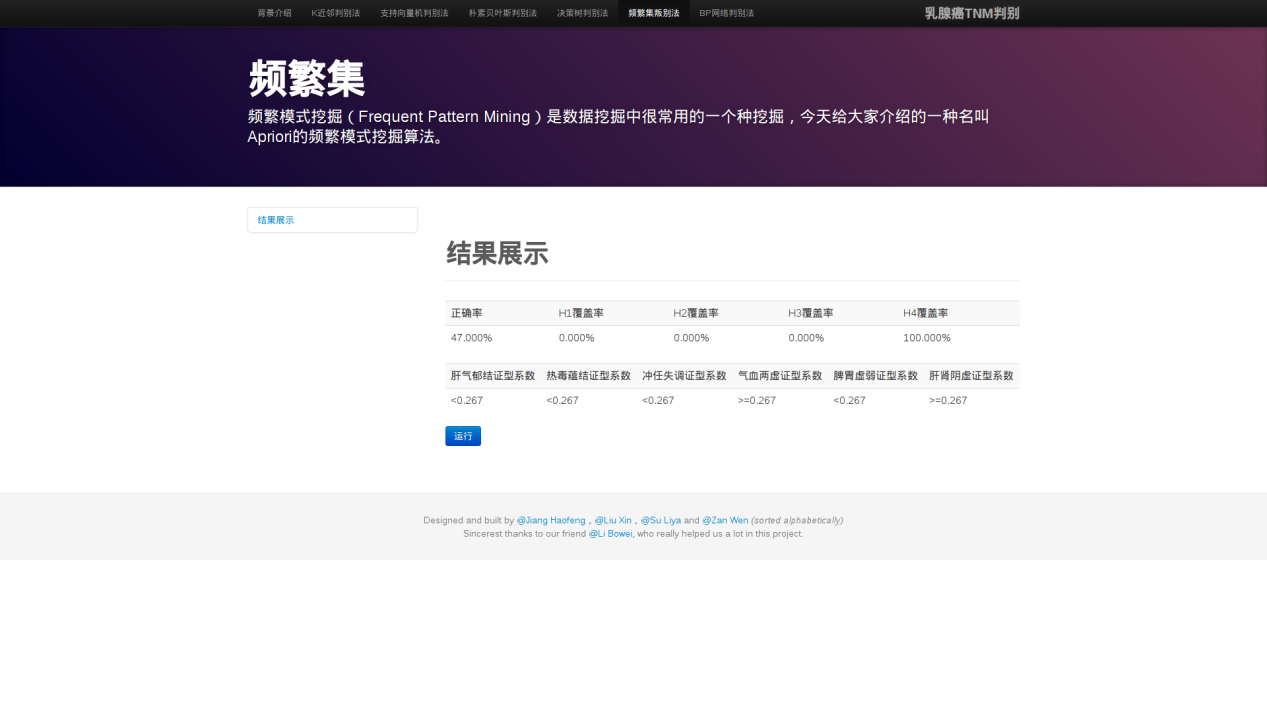


修改参数: g=20, c=30,训练svm模型并运行

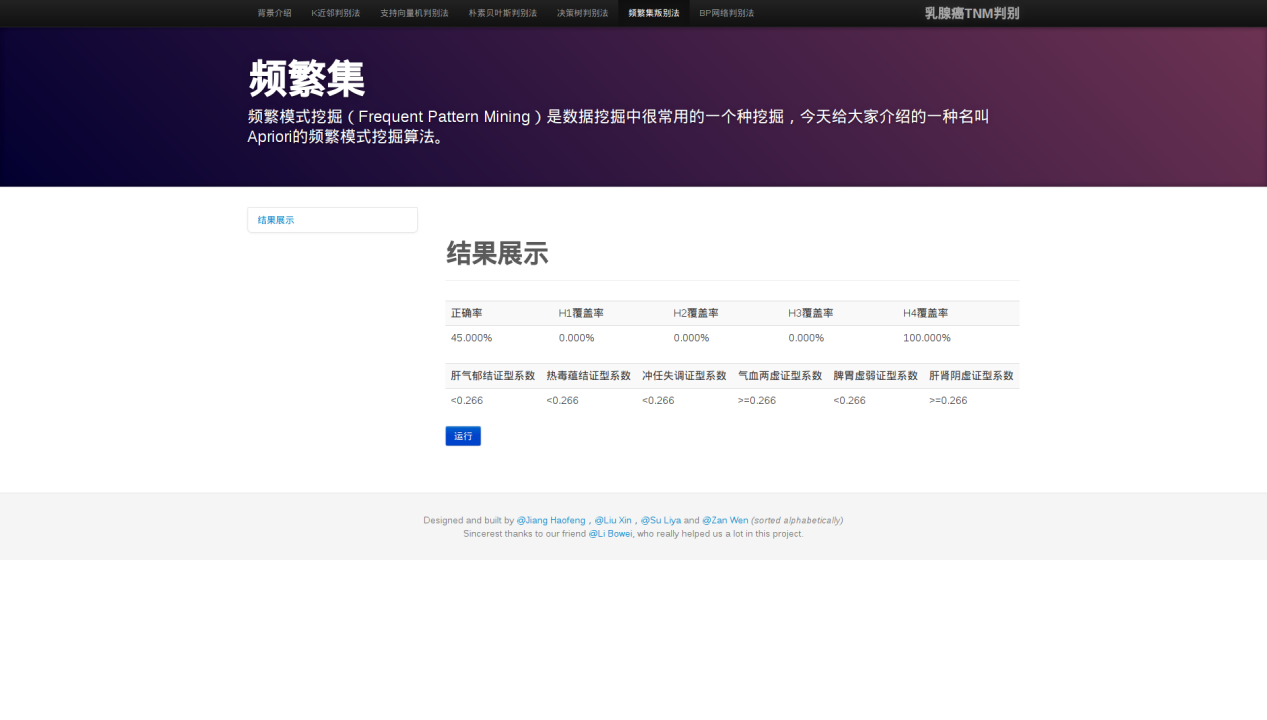


* 频繁集

首次运行结果



第二次运算结果



## 5.3实验结果分析：

分类算法分析：

1. BP神经网络：BP神经网络非常依赖于初始参数的设置，如果初始参数设置的比较偏离，需要通过上万次的迭代更新才能获得一般的结果，考虑结果显示的时间，并没有显示上万次迭代更新优化的结果，仅显示了百次的训练结果，所以结果并不是非常的好。同时，考虑样本数量有限，优化效果不明显的问题，并没有坚持优化BP神经网络。

2. 朴素贝叶斯：朴素贝叶斯基于变量的独立性给出了概率结果，通过实现发现效果仅在50%左右，结合实际信息样本中的变量关系关联性较强，并不符合朴素贝叶斯的基本假设。

3. 决策树：决策树的分支取决于属性的信息增益，对于离散数据的分类效果更好，并且具有很好的解释性。对于连续的数据，采用的思路主要是在每两个相邻的样本中间设置分割点，计算信息增益，找到最适合的分割点。决策树的实现不依赖于参数，比较固定没有学习和适应能力，在本实验中正确率仅在50%左右。

4. k近邻：k近邻的参数表明每个分类初始包含的项数，随着参数的增加，每类变得不够准确，导致最终的正确率下降。参数取2，2时由非常好的效果，正确率高达99%以上。说明乳腺癌疾病特征相似性的参数分类相似性关联很大。

5. 支持向量机：通过调整参数获得更大的最大边缘超平面，使得分类的准确率提高。通过参数调整，在50，100的情况下，准确率在96%以上。

6. 频繁集：

# 6.实验总结

通过乳腺癌病人病征数据的多种分类方法尝试，发现k近邻和支持向量机分类方法对于此类数据有较好的分类效果，有非常高的正确率和覆盖率，符合实验需求的要求。通过算法实现的过程，对于五种分类算法有了更加深入的了解与体会，特别是针对本实验中的数据为什么会出现响应的结果做了深入的探讨和分析。对于连续数据的频繁集查找做了一些尝试，深入体会了困难所在，以及一些初步的优化想法，例如通过信息增益来获得更好的离散化，通过增加关联关系去掉并非有趣的规则，以及多个规则中去掉冗余项。

项目已在github上公开（https://github.com/serea/DataMiningProject.git），后续针对数据挖掘频繁集以及工具化方面作进一步完善。