数据挖掘大作业

题目：乳腺癌病症的分析与建模

小组成员：

姓名：李嘉伟 学号：2016E8016061023

姓名：尹吉宪 学号：2016E8016061045

姓名：贺雨晴 学号：201628016029005

姓名：肖理想 学号：2016E8016061044

目录

目录.....I
1 简介.....1
2 数据预处理.....2
2.1 主要工作概述.....2
2.2 详细介绍.....2
2.2.1 对数据进行分析.....2
2.2.2 填补缺少的数值型数据.....3
2.2.3 对数值数据进行离散化.....5
2.2.4 填补缺少的非数值型数据.....8
3 挖掘关联信息.....11
3.1 关联信息.....11
3.1.1 挖掘频繁模式、关联和相关性.....11
3.1.2 基本概念.....11
3.2 Apriori 算法.....12
3.2.1 算法简介.....12
3.2.2 挖掘步骤.....12
3.2.3 基本概念.....12
3.2.4 实现步骤.....13
3.2.5 Apriori 算法所得的结果.....14
3.3 FP-tree 算法.....14
3.3.1 算法简介.....14
3.3.2 实现步骤.....15
3.3.3 FP-Tree 算法的结果.....18
4 结果分析.....19
4.1 主要工作概述.....19
4.2 详细介绍.....19

4.2.1 过滤关联规则.....19
4.2.2 分析得出结论.....19
4.2.3 更改置信度与支持度分析结果.....20
4.2.4 根据决策树分析结果.....21
5 总结.....24
6 小组分工明细.....25

1 简介

女性乳腺是由皮肤、纤维组织、乳腺腺体和脂肪组成的，乳腺癌是发生在乳腺上皮组织的恶性肿瘤。全球乳腺癌发病率自 20 世纪 70 年代末开始一直呈上升趋势。美国 8 名妇女中就会有 1 人患乳腺癌。中国不是乳腺癌的高发国家，但不宜乐观，近年我国乳腺癌发病率的增长速度高出发达国家 1～2 个百分点。乳腺癌已成为当前社会的重大公共卫生问题。

乳腺癌术后复发转移，是导致最终治疗失败的主要原因。中医药物治疗乳腺癌具有广泛的适应症和独特的优势。从整体出发，调整机体气压、阴阳、肺腑功能的平衡，根据不同的临床病症进行辨证论治。确定“先症而治”的方向：即在后续症状未出现之前，需要截断恶化病情的后续症状。发现中医症状间的关联关系和诸多症状之间的规律性，并且依据规则分析病因、预测病情的发展以及为未来临床提供有效借鉴。目前，中医的治疗一般采用中医辩证的原则，结合临床医师的经验和临床指南进行诊断，然而这种方法也存在一定的缺陷。首先，中医辨证极为灵活，虽然能处理患者复杂多变的临床症状，体现出治疗优势。但是缺乏统一的规范，难以做到诊断的标准化。其次，疾病的复杂性和体质的差异，造成病人是多种症素兼夹复合。临床医师可能会被自身的经验所误导，单纯对症治疗违背了中医辨证论治的原则。最后，统一症状的辨证分型，往往都有不同的见解。面对临床不同症状的患者，初学者难以判断。

中医治疗乳腺癌有优势也有缺陷。面对中医的缺陷，随着数据挖掘技术的发展，我们可以用数据挖掘技术对数据进行分析，得到中医症素与乳腺癌 TNM 分期之间的关系，弥补中医临床医师经验的缺陷，从而有助于中医对乳腺癌的治疗。

2 数据预处理

2.1 主要工作概述

该部分主要负责对原始数据进行预处理，低质量的数据将导致低质量的挖掘结果，因此对数据进行预处理是极其重要并且必不可少的工作。

主要工作分为以下几个方面：

1、对数据进行分析与整理

2、填补缺少的数值型数据

3、对数值数据进行离散化

4、填补缺少的非数值型数据

2.2 详细介绍

2.2.1 对数据进行分析

预处理数据时，首先要对数据进行分析，分析各数据的数据类型，即所给的数据中，每一个属性的数据类型，基本的数据类型包括：标称、二元、序数和数值类型。

通过对所给数据的分析，得出：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、肝肾阴虚证型系数均为数值类型的数据，而病程阶段、TNM 分期、转移部位、确诊后几年发现转移均为标称类型的数据。并且所给的数据中，对数值类型的数据均已经进行了规范化，映射到的区间为[0，1]。

观察数据发现，每一个属性的数据当中均存在缺失值。并且观察转移部位属性的数据发现，许多患者的记录当中，该属性包含多个值，因为每个人的癌细胞转移部位可以是一个器官，也可以是多个器官。因此，为了便于后续挖掘频繁项集的工作，可以将转移部位包含多个值的记录进行分离，即将形如表 2-1 的记录分离为形如表 2-2 的记录，这样进行数据挖掘的结果可以得到针对每个转移部位的频繁项集。

这样变换数据有助于挖掘不同分期阶段的乳腺癌患者的中医症素分布对转移部位的影响，但在挖掘与乳腺癌 TNM 分期之间的关系时，可以不对数据进

行这样的分离操作。只需留下 TNM 分期这一个标称属性与六个中医症素属性。将形如表 2-1 的记录转换为表 2-3 的记录。

表 2-1　未分离记录

0.242	0.280	0.131	0.210	0.191	0.351	S4	H4	R2R5	J1

表 2-2　分离记录

0.242	0.280	0.131	0.210	0.191	0.351	S4	H4	R2	J1
0.242	0.280	0.131	0.210	0.191	0.351	S4	H4	R5	J1

表 2-3　删除多余数据后的记录

0.242	0.280	0.131	0.210	0.191	0.351	H4

将所有数据处理成表 2-3 的形式后得到的数据存入“预处理数据及代码”文件夹的“1 after clean.xls”的“Sheet1”当中。

2.2.2　填补缺少的数值型数据

在该步骤，需要对缺少的数值类型数据进行填充，填充数据的方法有很多，可以使用属性的中心度量填充缺失值，也可以使用与给定元组属于同一类的所有样本的属性均值或中位数 进行填充。在这里仅针对缺少多个属性值的记录采用忽略元组的方法，其他记录均采用属性的中心度量填充缺失值。因为每条记录属于哪一类并不明确，数据当中可以代表类标号的属性有四个，即病程阶段、TNM 分期、转移部位、确诊后几年发现转移。

采用属性的中心度量填充缺失值时，需要对每个属性的数据进行分析，对于对称分布的数据，可以用均值来填充，而对于倾斜数据，则用中位数来填充，使用 python 程序对数据进行提取并分析，找出每个属性不同值出现的次数，然后得出其分布的图像，例如：属性肝气郁结证型系数的分布图像如图 2-1 所示：

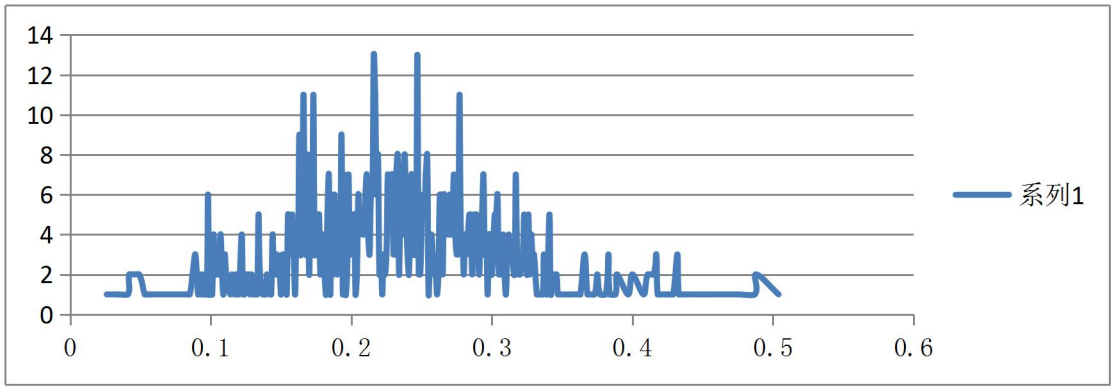

图 2-1　肝气郁结证型系数的分布图

而属性热毒蕴结证型系数的分布图像如图 2-2 所示：

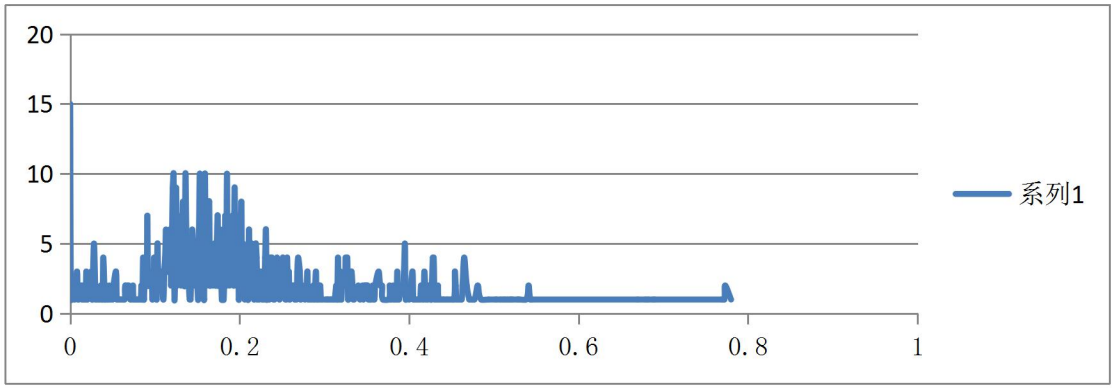

图 2-2　热毒蕴结证型系数的分布图

因此属性肝气郁结证型系数的空值可以用均值来替换，而热毒蕴结证型系数的空值则需要用中位数来替换，因其分布是倾斜的。同样根据分析发现，冲任失调证型系数、气血两虚证型系数、脾胃虚弱证型系数、肝肾阴虚证型系数均用均值来填补空缺值。

执行函数代码如下：

```
def operateData():
    data1=xlrd.open_workbook("in_file.xls")
    workbook=xlwt.Workbook()
    worksheet=workbook.add_sheet("sheet1")
    table=data1.sheets()[0]      #获取一个工作表
    colnum=6
    rownum=table.nrows
    for i in range(colnum):
        L=table.col_values(i)
        #del L[0]
        ave=average(L) #求平均值函数
```

```
        for j in range(rownum):
            if L[j]=='':
                L[j]=ave
            worksheet.write(j,i,L[j])
    workbook.save("out_file.xls")
```

将处理后的数据存入“预处理数据及代码”文件夹的“2 fill num blanks.xls”的“Sheet1”表中。

2.2.3 对数值数据进行离散化

在所给数据中，所有的数值型数据均已规范化，但是如果不对其进行处理而直接进行频繁模式挖掘，则挖掘出的知识会很难理解并且非常复杂，主要原因便是存在大量的数字，而 Apriori 算法无法处理连续型数值变量。因此，需要对数据进行规约，也就是对每个属性的数据进行离散化，使用一个标签来对应一个区间。在这里采用无监督学习的方法对数据进行离散化，所使用的算法是 K-means，采用 C++来实现，时间复杂度为 O(nkI)，其中 n 为需要聚类的对象的总数，k 为簇数，I 是迭代的次数。采用 vector<>来代替 C 语言中的数组，vector<>可以不指定空间大小自动生长，并且有很多可用的方法，因此采用 vector<>将方便程序对数据的处理。但是采用 K-means 算法有很多缺陷，聚类结果受到初始聚类中心、K 值的影响，因此需要对这两方面进行相应的改进。

1、初始聚类中心的选取

初始聚类中心的选取：对每一个属性进行 K-means 聚类，算法是在一维空间上运行的，并且属性值均是数值类型，可以首先找到选定属性的最大值 max 和最小值 min，定义 step 为步长，step=(max-min)/K，根据 min，max，step 得到初始聚类中心为{min，min+1*step，min+2*step，…，max}，然后迭代进行求解，这样选择将比随机选择初始聚类中心所得到的结果要更好一些。

2、K 值的选择

第二个问题便是 K 值的选择，通常，事先并不知道给定的数据集应该分多少类才是合适的，但是 K 在一般的情况下满足 $2\leq K_{opt}\leq\sqrt{n}$，其中 K_{opt} 是最优值。可以利用已有的聚类评价方法来选取最优的 K 值，K 的值也不应该太大，因为数据已经规范化到了[0，1]区间，如果分得太细，反而不利于挖掘，并且 K 值过大会导致某些区间的数据量太少，不具备代表性。在这里选用两个度量指标来评价聚类的质量，分别为绝对误差标准和轮廓系数。

绝对误差的表达式为：$\sum_{i=1}^{k}\sum_{p\in Cj} dist(p,o_i)$，该值所表达的含义是每个样本到其所属类簇的中心的距离的和，因此如果每个簇越紧密，聚类效果越好，该值会变小，并且，当我们尝试的 K 的值小于 K_{opt} 时，该指标会急速下降，而当我们尝试的 K 的值等于或者高于真实的类簇的数目时，该指标下降得会很缓慢，因此根据图像中落差最大的点，即有明显拐点的地方，便可以近似得出 K 的最优值 K_{opt}。上述方法便是肘方法。

第二个标准便是轮廓系数，Silhouette 系数是对聚类结果有效性的解释和验证，基本方法如下：

(1) 计算样本 i 到所属簇内其他样本的距离的平均值 a_i；

(2) 计算样本 i 到其他簇 C_j 的所有样本的距离的平均值 b_{ij}，然后选取最小值，即 $b_i=\min\{b_{i1}, b_{i2}, \cdots, b_{ik}\}$；

(3) 样本 i 的轮廓系数定义为 S(i)=(b(i)-a(i))/max{a(i),b(i)}；

(4) 将所有样本点的轮廓系数求平均，就是该聚类结果的总的轮廓系数。

同样，轮廓系数也可以用在肘方法当中。可以发现，s(i)所在的区间是[-1，1]，s(i)接近 1，则说明样本 i 聚类合理；s(i)接近-1，则说明样本 i 更应该分类到另外的簇；若 s(i) 近似为 0，则说明样本 i 在两个簇的边界上。

因此整体的轮廓系数的值越大，越接近于 1，说明分类的效果越好。我们可以通过尝试不同的 K，并观察不同的 K 与绝对误差和轮廓系数的关系图，找到能得到最优的分类结果的 K 值。

以冲任失调证型系数为例：

通过尝试不同的 K 值，得到的 K 与绝对误差的关系图如图 2-3 所示：

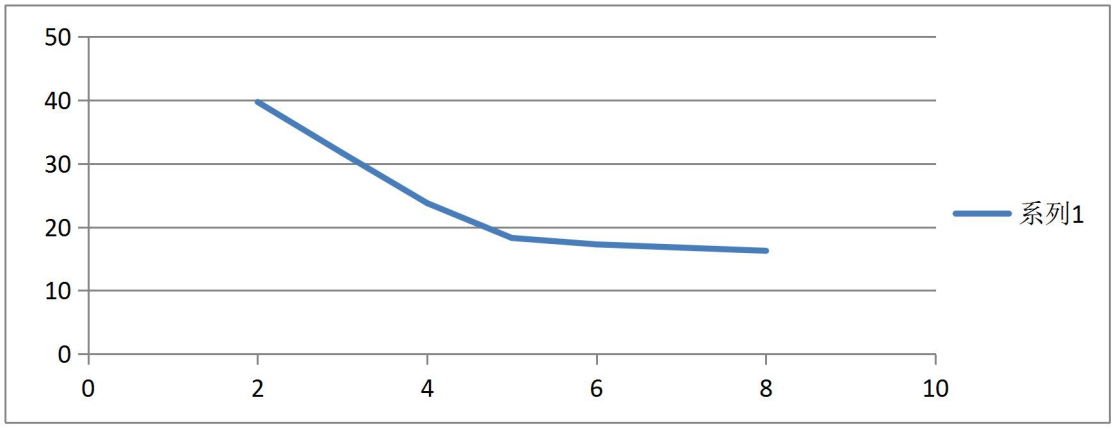

图 2-3　冲任失调证型系数绝度误差走向图

得到的 K 与轮廓系数的关系图如图 2-4 所示：

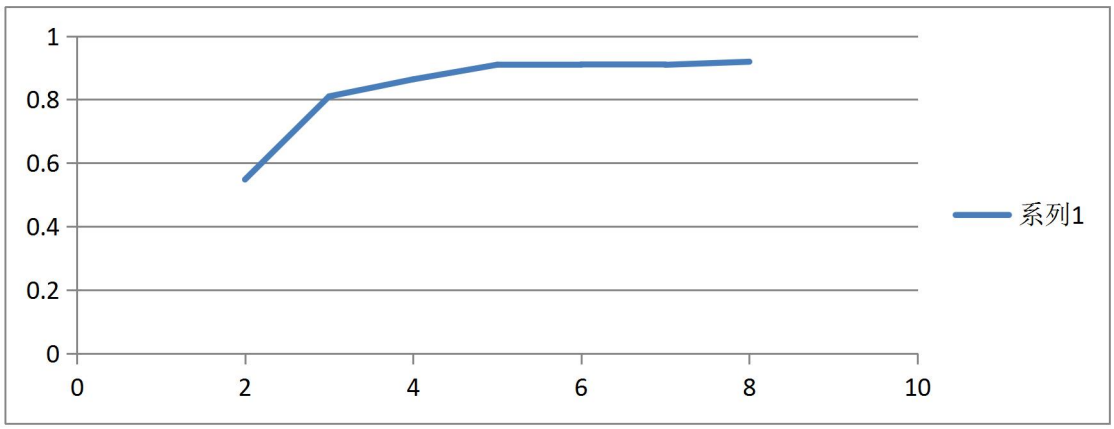

图 2-4　冲任失调证型系数轮廓系数走向图

可以发现在 K=5 时，有明显拐点，并且在 K=6 时，绝对误差下降并不明显，对于轮廓系数而言，同样在 K=5 之后，上升趋势也不明显，因此 K 的最优值 K_{opt}=5，即将冲任失调证型系数分为 5 类。

使用同样的方法，将各属性进行聚类然后进行离散化处理，使得每一个数据有一个对应的标签。分类结果如表 2-4~表 2-9 所示：

表 2-4　肝气郁结证型系数

标号	区间范围	元素个数
A1	(0.000,0.137]	100
A2	(0.137,0.203]	225
A3	(0.203,0.266]	301
A4	(0.266,0.355]	253
A5	(0.355,0.504]	50

表 2-5　热毒蕴结证型系数离散表

标号	区间范围	元素个数
B1	(0.000,0.134]	267
B2	(0.134,0.238]	370
B3	(0.238,0.364]	154
B4	(0.364,0.545]	128
B5	(0.545,0.780]	10

表 2-6　冲任失调证型系数离散表

标号	区间范围	元素个数
C1	(0.000,0.184]	231
C2	(0.185,0.250]	295
C3	(0.250,0.317]	247
C4	(0.317,0.415]	121
C5	(0.415,0.610]	35

表 2-7　气血两虚证型系数离散表

标号	区间	元素个数
D1	(0.000,0.166]	262
D2	(0.166,0.231]	321
D3	(0.231,0.297]	216
D4	(0.297,0.382]	101
D5	(0.382,0.552]	29

表 2-8　脾胃虚弱证型系数离散表

标号	区间范围	元素个数
E1	(0.000,0.155]	283
E2	(0.155,0.259]	313
E3	(0.259,0.378]	245
E4	(0.378,0.526]	88

表 2-9　肝肾阴虚证型系数离散表

标号	区间	元素个数
F1	(0.000,0.178]	200
F2	(0.178,0.261]	237
F3	(0.261,0.353]	264
F4	(0.353,0.607]	228

将处理的数据存入“预处理数据及代码”文件夹的“3 dicrete data.xls”的“Sheet1”当中。

2.2.4　填补缺少的非数值型数据

非数值型数据，在这里即标称数据，分别为病程阶段、TNM 分期、转移部位、确诊后几年发现转移，这几个属性也可以看为由肝气郁结证型系数、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、

肝肾阴虚证型系数推出的结论，并且以上各数值属性都是相互独立的，因此在填这三个标称属性的空值时，可以用朴素贝叶斯分类的方法，每个标称属性的值可以看作是一个类标号，利用经过离散化后的数据来对空缺值进行预测，相当于监督学习中的分类预测，使用 python 程序进行实现。

例：填 TNM 分期的空缺值，则每条记录的类标号就是其对应的 TNM 分期的属性值，找到空缺的记录，例：表 1=10 的记录：

表 2-10　记录

A2	B1	C2	D3	E3	F3	S2		R1	J3

该记录的 TNM 分期值空缺，根据贝叶斯方法对其进行预测：

X=(肝气郁结证型系数=A2，热毒蕴结证型系数=B1，冲任失调证型系数=C2，气血两虚证型系数=D3，脾胃虚弱证型系数=E3，肝肾阴虚证型系数=F3)，

对记录 X 进行预测：

P(TNM 分期=H1)=0.092348　　P(TNM 分期=H2)=0.179419

P(TNM 分期=H3)=0.182099　　P(TNM 分期=H4)=0.547933

P(肝气郁结证型系数=A2|TNM 分期=H1)=0.276083

P(热毒蕴结证型系数=B1|TNM 分期=H1)=0.330658

P(冲任失调证型系数=C2|TNM 分期=H1)=0.343499

P(气血两虚证型系数=D3|TNM 分期=H1)=0.335473

P(脾胃虚弱证型系数=E3|TNM 分期=H1)=0.351524

P(脾胃虚弱证型系数=F3|TNM 分期=H1)=0.205457

P(X| TNM 分期=H1)= P(肝气郁结证型系数=A2|TNM 分期=H1)

×P(热毒蕴结证型系数=B1|TNM 分期=H1)

×P(冲任失调证型系数=C2|TNM 分期=H1)

×P(气血两虚证型系数=D3|TNM 分期=H1)

×P(脾胃虚弱证型系数=E3|TNM 分期=H1)

×P(脾胃虚弱证型系数=F3|TNM 分期=H1)

=0.000235

同理，计算出：

P(X| TNM 分期=H2)=0.000125

P(X| TNM 分期=H3)=0.000151

P(X| TNM 分期=H4)=0.000793

为了便于比较，将结果都乘以 10000，然后进行比较：

P(X| TNM 分期=H1)P(TNM 分期=H1)×10000=0.217280

P(X| TNM 分期=H2)P(TNM 分期=H1)×10000=0.224457

P(X| TNM 分期=H3)P(TNM 分期=H2)×10000=0.273736

P(X| TNM 分期=H4)P(TNM 分期=H3)×10000=4.349698

因此 X 的 TNM 分期的预测值为 T4。

按照上述方法，对 TNM 分期的缺失值进行预测，得到结果，并将预测数据回填入原始数据中，得到最终的结果，存入“预处理数据及代码”文件夹的“4 fill all blanks.xls”的“Sheet1”中。

至此，所有的数据均处理完毕，最终的需要进行挖掘的数据为“4 fill all blanks.xls”的“Sheet1”表中的数据。

3 挖掘关联信息

3.1 关联信息

3.1.1 挖掘频繁模式、关联和相关性

频繁模式是频繁地出现在数据集中的模式（如项集、子序列或子结构）。例如，频繁地同时出现在交易数据集中的商品（如牛奶和面包）的集合是频繁项集。一个子序列，如首先购买 pc，然后是数码相机，再后是内存卡，如果它频繁地出现在购物历史数据库中，则称它为一个频繁的序列模式。对于挖掘数据之间的关联、相关性和许多其他有趣的联系，发现这种频繁模式起着至关重要的作用。此外，它对数据分类、聚类和其他数据挖掘任务也有帮助。因此，频繁模式的挖掘就成了一项重要的数据挖掘任务和数据挖掘研究关注的主题之一。

3.1.2 基本概念

频繁模式是频繁出现在数据集中的模式（如项集，子序列和子结构）。频繁模式可以用关联规则表示。如何判断模式是否频繁，有两个基本的度量：

支持度：该模式在所有被考察的对象中的占比，表示了该模式（规则）的有用性。

置信度：由规则的前因推出后果的可信度，表示了规则的确定性。

设规则为 A->B，则支持度和置信度可以表示如下：

support(A->B) = P(AUB)

confidence(A->B) = P(A|B)

根据上面的定义，可以得出挖掘关联规则“A->B”的问题可以归结为挖掘频繁项集（因为这里的概率运算都可以用满足条件的项的出现次数和总个数的比表示）：

(1) 找出所有的频繁项集。

(2) 有频繁项集产生强关联规则。

将可以看到，第一步的开销远大于第二步，所以性能将由第一步决定。

挖掘海量数据的主要挑战是这种挖掘常常产生大量满足最小支持度阈值的项集，因为如果一个项集是频繁的，那么它的每个子集也是频繁的。比如说包

含 100 个项集的子集就远远超出了目前计算机的储存限制。为了克服这个问题，提出了比频繁项集和极大频繁项集的概念。

闭频繁项集：如果不存在真超项集 Y 使得 Y 与 X 在项集 S 中有相同的支持度计数，则称 X 在 S 中是闭的。项集 X 是 S 中的闭频繁项集，如果 X 在 S 中是闭的和频繁的；

极大频繁项集：如果 X 是频繁的，并且不存在超项集 Y 使得 X 属于 Y，并且 Y 在 S 中是频繁的。

闭频繁项集的集合包含了频繁项集的完整信息，但是从极大频繁项集只能知道一个特定的项集是否是频繁的，而无从得知其实际支持度计数。

Apriori 算法是一种发现频繁项集的基本算法。接下来我们将用 Apriori 算法来挖掘题目中的关联信息，并且根据 Apriori 算法的缺陷，用 FP-tree 算法重新挖掘题目中的关联信息，这也同时是对 Apriori 算法的验证，确保我们挖掘的信息是有用的、正确的。

3.2 Apriori 算法

3.2.1 算法简介

Apriori 算法是一种挖掘频繁项集的算法。Apriori 算法应用广泛，可用于消费市场价格分析，猜测顾客的消费习惯；网络安全领域中的入侵检测技术；可用在高校管理中，根据挖掘规则可以有效地辅助学校管理部门有针对性地开展贫困助学工作；也可用在移动通信领域中，指导运营商的业务运营和辅助业务提供商的决策制定。

3.2.2 挖掘步骤

(1) 依据支持度找出所有频繁项集（频度）

(2) 依据置信度产生关联规则（强度）

3.2.3 基本概念

对于 A->B

支持度：P(A∩B)，既有 A 又有 B 的概率

置信度：P(B|A)，在 A 发生的同时，B 发生的概率

如果事件 A 中包含 k 个元素，那么称这个事件 A 为 k 项集事件 A 满足最小支持度阈值的事件称为频繁 k 项集。同时满足最小支持度阈值和最小置信度阈值的规则称为强规则。

3.2.4 实现步骤

Apriori 算法是一种最有影响的挖掘频繁项集的算法。Apriori 使用一种称作逐层搜索的迭代方法，“K-1 项集”用于搜索“K 项集”。

首先，找出频繁“1 项集”的集合，该集合记作 L1。L1 用于找频繁“2 项集”的集合 L2，而 L2 用于找 L3。如此下去，直到不能找到“K 项集”。找每个 Lk 都需要一次数据库扫描。

核心思想是：连接步和剪枝步。连接步是自连接，原则是保证前 k-2 项相同，并按照字典顺序连接。剪枝步，是使任一频繁项集的所有非空子集也必须是频繁的。反之，如果某个候选的非空子集不是频繁的，那么该候选肯定不是频繁的，从而可以将其从 Ck 中删除。

简单的讲，发现频繁项集过程为：

(1) 扫描、计数、比较、产生频繁项集、连接、剪枝、产生候选项集。重复上述步骤直到不能发现更大的频繁项集。

(2) 产生关联规则，过程为：根据前面提到的置信度的定义，关联规则的产生如下：

a.对于每个频繁项集 L，产生 L 的所有非空子集；

b.对于 L 的每个非空子集 S，如果 P(L)/P(S)≧min_conf，则输出规则“L->S”。

生成规则的执行函数的代码：

```
def generateRulesPlus(L,supportData, minConf=0.75):
    bigRuleList = []
    for H1 in L[1]:

        for h1 in H1:
            h1 = frozenset([h1])
            conf = supportData[H1] / supportData[h1]
            if conf >= minConf:
                bigRuleList.append(((h1,supportData[h1]),(H1-
h1,supportData[H1-h1]),conf,(h1,'-->',H1-h1)))
    for H1 in L[2]:
```

```
        for h1 in H1:
            h1 = frozenset([h1])
            conf = supportData[H1] / supportData[h1]
            if conf >= minConf:
                bigRuleList.append(((h1,supportData[h1]),(H1-
h1,supportData[H1-h1]),conf,(h1,'-->',H1-h1)))
            conf = supportData[H1] / supportData[H1-h1]
            if conf >= minConf:
                bigRuleList.append(((H1-h1,supportData[H1-
h1]),(h1,supportData[h1]),conf,(H1-h1,'-->',h1)))

    return bigRuleList
```

3.2.5 Apriori 算法所得的结果

所得结果如下所示：

D3∧F4=>H4	support = 0.73210	confidence = 0.86765
A4∧F4=>H4	support = 0.08073	confidence = 0.86867
C4∧F4=>H4	support = 0.06674	confidence = 0.83871
A3∧E3=>B2	support = 0.09795	confidence = 0.86056
E3∧F3=>H4	support = 0.06566	confidence = 0.80328
B2∧F4=>H4	support = 0.07750	confidence = 0.79167
A3∧E3=>H4	support = 0.09795	confidence = 0.76923
A3∧E4=>H4	support = 0.07643	confidence = 0.76056
A3∧E3=>B2	support = 0.09795	confidence = 0.86056
D2∧F3∧H4=>A2	support = 0.08795	confidence = 0.76056

结果的后续处理与分析在第四部分。

3.3 FP-tree 算法

3.3.1 算法简介

FP-tree 算法的产生得益于 Apriori 算法的两个缺陷。在许多情况下，Apriori 算法的候选产生——检查方法显著压缩了候选项集的规模，并产生了很好的性能，然而它可能受两种非平凡开销的影响：

(1) 它可能仍然需要大量候选项集。例如，如果有 10^4 个频繁 1 项集，则 Apriori 算法需要产生多大 10^7 个候选 2 项集。

(2) 它可能需要重复地扫描整个数据库，通过模式匹配检查一个很大的候选集合。检查数据库中每个事务来确定候选项集支持度的开销很大。

由此产生了 FP-tree 算法。它采取如下分治策略：首先，将代表频繁项集的数据库压缩到一棵频繁模式树，该树仍保留项集的关联信息。然后，把这种压缩后的数据库划分成一组条件数据库（一种特殊类型的投影数据库），每个数据库关联一个频繁项或“模式段”，并分别挖掘每个条件数据库。对于每个模式片段，只需要考察与它相关联数据集。因此，随着被考察的模式的增长，这种方法可以显著的压缩被搜索的数据集的大小。

3.3.2 实现步骤

FP-tree 算法由构造 FP 树和挖掘 FP 树两部分构成。

构造 FP 树：

(1) 扫描事务库 D，获得 D 中所包含的全部频繁项集 1F，及它们各自的支持度。对 1F 中的频繁项按其支持度降序排序得到 L。

(2) 创建 FP-tree 的根结点 T，以“null”标记。再次扫描事务库。对于 D 中每个事务，将其中的频繁项选出并按 L 中的次序排序。设排序后的频繁项表为[p|P]，其中 p 是第一个频繁项，而 P 是剩余的频繁项。调用 insert_tree([p|P],T)，insert_tree([p|P],T)过程执行情况如下：如果 T 有子女 N 使 N .item_name= p.item_name，则 N 的计数增加 1；否则创建一个新结点 N，将其计数设置为 1，链接到它的父结点 T，并且通过 node_link 将其链接到具有相同 item_name 的结点。如果 P 非空，递归地调用 insert_tree(P，N)。FP-tree 是一个高度压缩的结构，它存储了用于挖掘频繁项集的全部信息。FP-tree 所占用的内存空间与树的深度和宽度成比例，树的深度一般是单个事务中所含项目数量的最大值；树的宽度是平均每层所含项目的数量。由于在事务处理中通常会存在着大量的共享频繁项，所以树的大小通常比原数据库小很多。频繁项集中的项以支持度降序排列，支持度越高的项与 FP-tree 的根距离越近，因此有更多的机会共享结点，这进一步保证了 FP-tree 的高度压缩。

如图 3-1 所示：

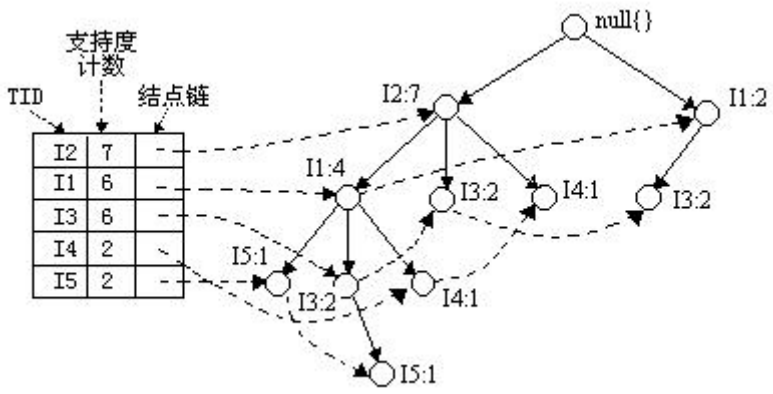

图 3-1　FP 树

另外还要交代一下 FPTree 算法中几个基本的概念：

FP-Tree：是把事务数据表中的各个事务数据项按照支持度排序后，把每个事务中的数据项按降序依次插入到一棵以 NULL 为根结点的树中，同时在每个结点处记录该结点出现的支持度。

条件模式基：包含 FP-Tree 中与后缀模式一起出现的前缀路径的集合。也就是同一个频繁项在 FP 树中的所有节点的祖先路径的集合。

我们来模拟一下 FP-growth 的执行过程：

1、 在 FP-growth 递归调用的第一层，模式前后 a=NULL，得到的其实就是频繁 1-项集。

2、 对每一个频繁 1-项，进行递归调用 FP-growth()获得多元频繁项集。

下面举个例子说明 FP-growth 的执行过程，I5 的条件模式基是(I2 I1:1)，(I2 I1 I3:1)，I5 构造得到的条件 FP-Tree 如图 3-2 所示。然后递归调用 FP-growth，模式后缀为 I5。这个条件 FP-Tree 是单路径的，在 FP_growth 中直接列举{I2:2，I1:2，I3:1}的所有组合，之后和模式后缀 I5 取并集得到支持度大于 2 的所有模式：{ I2 I5:2, I1 I5:2, I2 I1 I5:2}。

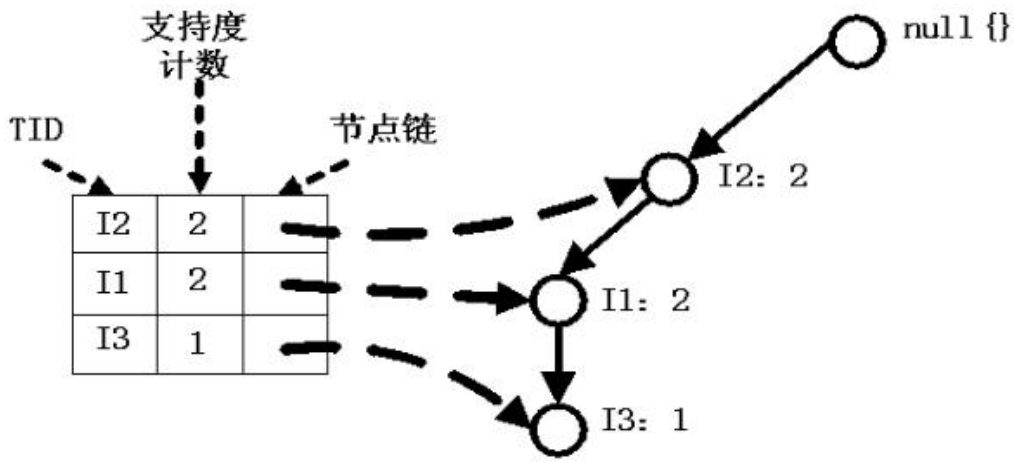

图 3-2 条件树

挖掘 FP 树：

procedure FP_growth(*Tree*, *a*)

if *Tree* 含单个路径 *P* **then{**

for 路径 *P* 中结点的每个组合（记作 *b*）

产生模式 *b* ∪ *a*，其支持度 *support* = *b* 中结点的最小支持度；

} else {

for each a_i 在 *Tree* 的头部(按照支持度由低到高顺序进行扫描){

产生一个模式 $b = a_i \cup a$，其支持度 $support = a_i.support$；

构造 *b* 的条件模式基，然后构造 *b* 的条件 FP-树 *Tree*b；

if *Tree*b 不为空 then

调用 FP_growth (*Tree*b, *b*)；

}

}

FP-growth 函数的输入：tree 是指原始的 FPTree 或者是某个模式的条件 FPTree，a 是指模式的后缀（在第一次调用时 a=NULL，在之后的递归调用中 a 是模式后缀）。

FP-growth 函数的输出：在递归调用过程中输出所有的模式及其支持度（比如{I1,I2,I3}的支持度为 2）。每一次调用 FP_growth 输出结果的模式中一定包含 FP_growth 函数输入的模式后缀。

3.3.3 FP-Tree 算法的结果

由于在使用 FP-Tree 算法时，只是挖掘频繁项集，并没有在其中置入置信度的约束，因此所得到的结果是所有的频繁 1 项集、2 项集...，然后再调用求解置信度的函数对所有的频繁项集进行置信度计算，得到最终符合要求的关联规则。

频繁项集挖掘函数：

```
def mineTree(inTree, headerTable,minSup,preFix,freqItemList):
    bigL = [v[0] for v in headerTable.items()]
    #print(bigL)
    for basepat in bigL:
        newFreqSet = preFix.copy()
        newFreqSet.add(basepat)
        freqItemList.append(newFreqSet)
        condPatBases = findPrefixPath(basepat,headerTable[basepat][1])
        myCondTree, myHead = createTree(condPatBases,minSup)
    if myHead !=None:
        mineTree(myCondTree,myHead,minSup,newFreqSet,freqItemList)
```

FP-Tree 算法所得到的频繁项集有很多，这里列出部分结果，完整结果存于“挖掘算法”文件夹的“FP-Tree result.docx”文件中。

频繁 1 项集：

frozenset({'F4'})

frozenset({'H4'})

frozenset({'A4'})

频繁 2 项集：

frozenset({'A4', 'F1'})

frozenset({'A3', 'C2'})

frozenset({'D2', 'E2'})

频繁 3 项集：

frozenset({'A3', 'B2', 'F3'})

frozenset({'B2', 'D2', 'H4'})

frozenset({'A3', 'E2', 'H4'})

在频繁项集当中，对每一个频繁项集进行置信度的计算，得到满足最小置信度的关联规则。

4 结果分析

4.1 主要工作概述

该部分所做的主要工作为对挖掘的结果进行处理和分析，并得出最终的结论。因为很多结果并不是有意义的，我们所要找的有用的关联规则为在满足最小支持度和置信度的前提下，可以由中医症素推出乳腺癌 TNM 分期的规则。

主要工作为：

1、过滤关联规则

2、分析得出结论

3、更改置信度与支持度分析结果

4、根据决策树分析结果

4.2 详细介绍

4.2.1 过滤关联规则

我们所要考察的是中医症素与乳腺癌 TNM 分期之间的关系，因此只需要关注那些以 TNM 分期作为结果的规则。例如：F3=>H4 这样的规则。因此在最小支持度为 6%并且最小置信度为 75%的情况下，得到的有用的关联规则如下：

D3∧F4=>H4	support = 0.73210	confidence = 0.86765
A4∧F4=>H4	support = 0.08073	confidence = 0.86867
C4∧F4=>H4	support = 0.06674	confidence = 0.83871
E3∧F3=>H4	support = 0.06566	confidence = 0.80328
B2∧F4=>H4	support = 0.07750	confidence = 0.79167
A3∧E3=>H4	support = 0.09795	confidence = 0.76923
A3∧E4=>H4	support = 0.07643	confidence = 0.76056

以上是在满足最小支持度和置信度条件下的关联规则。

4.2.2 分析得出结论

根据得出的关联规则，我们可以得出以下的结论：

(1) A4∧F4=>H4：从关联规则中可以发现，A4∧F4=>H4 的支持度为 8.073%，置信度为 86.867%，说明肝气郁结证型系数在(0.266，0.355]区间内，肝肾阴虚证型系数在(0.353，0.607]区间内，TNM 分期被诊断为 H4 的可能性为 86.867%，这种情况发生的可能性为 8.073%，并且这种情况的置信度达到了最大。

(2) A3∧E3=>H4：从关联规则中可以发现，A3∧E3=>H4 的支持度为 9.795%，置信度为 76.923%，说明肝气郁结证型系数在(0.266，0.355]区间内，脾胃虚弱证型系数在(0.259，0.378]区间内，TNM 分期被诊断为 H4 的可能性为 76.923%，这种情况发生的可能性为 9.795%，并且这种情况的支持度达到了最大。

(3) 以上是支持度与置信度最大的两条规则，分析其他规则，发现在所有的关联规则当中，大部分都与肝肾阴虚证型系数、肝气郁结证型系数以及脾胃虚弱证型系数有着密切的关系，这三项指标均很高，其中肝肾阴虚证型系数均在 0.353 以上，肝气郁结证型系数均在 0.203 以上，脾胃虚弱证型系数均在 0.259 以上。

因此，综合分析可以得出，TNM 分期已经达到 H4 的患者，其主要的表现症状为肝肾阴虚证、肝气郁结证以及脾胃虚弱证，并且在这三项指标中，患者的肝肾阴虚证和肝气郁结证的表现尤为突出，置信度达到 86.867%，因此这两项指标是对患者检查的重要指标。

4.2.3 更改置信度与支持度分析结果

目前设置的置信度为 75%，为了更好得发现中医症素与乳腺癌 TNM 分期之间的关系，可以适当降低置信度，来观察所得出模型的结果。因此将置信度改为 70%进行实验操作。得出的过滤之后的关联规则如下：

D3∧F4=>H4	support = 0.07319	confidence = 86.76470
A4∧F4=>H4	support = 0.08073	confidence = 86.66667
C4∧F4=>H4	support = 0.06673	confidence = 83.87094
E3∧F3=>H4	support = 0.06566	confidence = 80.32786
B2∧F4=>H4	support = 0.07750	confidence = 79.16666
A3∧E3=>H4	support = 0.09795	confidence = 76.92307

A3∧F4=>H4	support = 0.07642	confidence = 76.05630
F4=>H4	support = 0.24219	confidence = 73.77777
A4∧C4=>H4	support = 0.06027	confidence = 73.21427
A3∧D2∧F3=>H4	support = 0.07965	confidence = 71.62162
D2∧F4=>H4	support = 0.08073	confidence = 70.66666

从以上的关联规则中可以发现 TNM 分期已经达到 H4 的患者，很明显，大部分的人的肝肾阴虚证型系数和肝肾阴虚证系数都很高。

4.2.4 根据决策树分析结果

根据以上的频繁项集的挖掘，我们可以发现 TNM 分期为 H4 的患者有很多，并且也是我们所重点关注的数据，因此挖掘出的频繁项集基本上都是与 H4 分期相关的，但是为了得出中医症素与其他 TNM 分期之间的关系，使用 Apriori 或者 FP-Growth 将不再可靠，因为只有大幅降低置信度或者抬高支持度才可以出现结果为 H1、H2 或者 H3 的频繁项集。

因此，在预处理后的数据集上进行决策树模型的构建，便可以方便得得出各中医症素与 TNM 分期之间的关系，在这里使用信息增益作为属性选择度量，对 ID3 算法进行实现。

在这里使用 python 编写的决策树程序，对预处理后的数据进行处理，使用 python 自带的画图工具，画图函数代码如下：

```
def createPlot(inTree):
    fig = plt.figure(1, facecolor='white')
    fig.clf()
    axprops = dict(xticks=[], yticks=[])
    createPlot.ax1 = plt.subplot(111, frameon=False, **axprops)
    plotTree.totalW = float(getNumLeafs(inTree)) #树的宽度
    plotTree.totalD = float(getTreeDepth(inTree)) #树的深度
    plotTree.xOff = -0.5/plotTree.totalW; plotTree.yOff = 1.0;
    plotTree(inTree, (0.5,1.0), '')
    plt.show()
```

得出最后的决策树模型如图 4-1 所示：

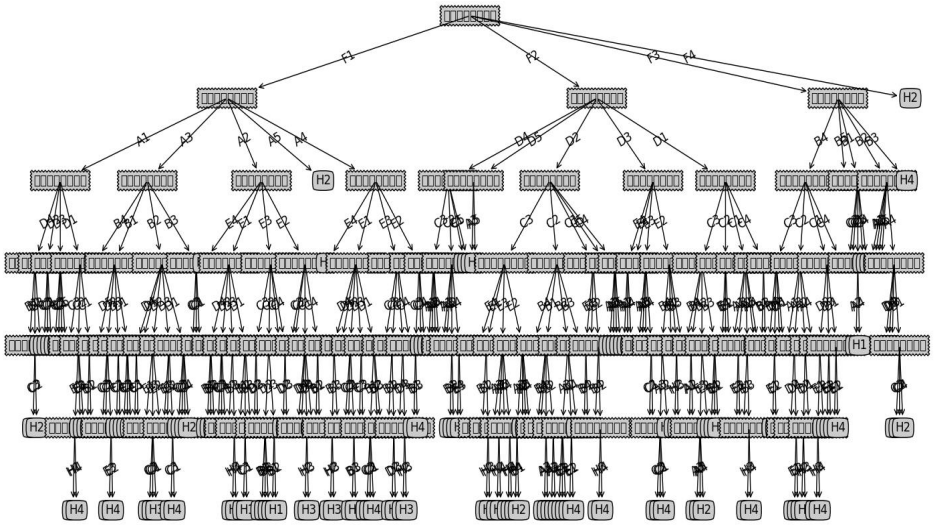

图 4-1　决策树模型

但是得出的决策树模型难以进行观察，如果选的数据量少一些，作为训练集，在这里随机选取 100 条数据进行训练，则可以得出如图 4-2 所示的决策树模型，在这里的欠缺之处为很难通过图上得出结论，可视性比较低，因此直接在程序中进行规则的输出：

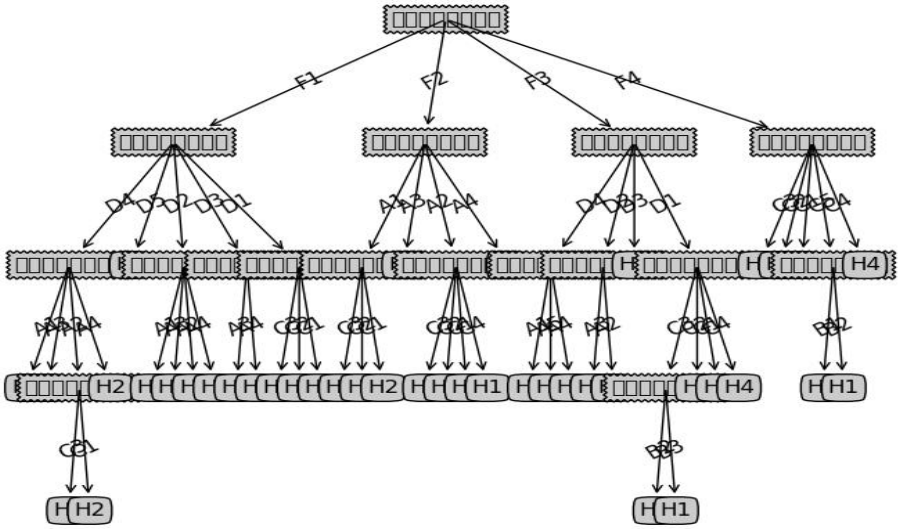

图 4-2　100 条数据决策树模型

根据决策树得到的规则如下：

if 肝肾阴虚证型系数 = F2 and 肝气郁结证型系数 = A1 then TNM=H1

if 肝肾阴虚证型系数 = F2 and 肝气郁结证型系数 = A2 and 热毒蕴结证型系数 =B2 then TNM = H1

if 肝肾阴虚证型系数 = F1 and 热毒蕴结证型系数 = B1 and 冲任失调证型系数 = C3 then TNM = H2

if 肝肾阴虚证型系数 = F1 and 热毒蕴结证型系数 = B2 then TNM = H2

if 肝肾阴虚证型系数 = F2 and 肝气郁结证型系数 = A2 and 热毒蕴结证型系数 = B1 then TNM = H2

if 肝肾阴虚证型系数 = F1 and 热毒蕴结证型系数 = B3 then TNM = H3

if 肝肾阴虚证型系数 = F2 and 肝气郁结证型系数 = A3 then TNM = H3

if 肝肾阴虚证型系数 = F2 and 肝气郁结证型系数 = A5 then TNM = H3

if 肝肾阴虚证型系数 = F1 and 热毒蕴结证型系数 = B1 and 冲任失调证型系数 = C1 then TNM = H4

if 肝肾阴虚证型系数 = F2 and 肝气郁结证型系数 = A2 and 热毒蕴结证型系数 = B3 then TNM = H4

if 肝肾阴虚证型系数 = F2 and 肝气郁结证型系数 = A3 and 脾胃虚弱证型系数 = E3 then TNM = H4

if 肝肾阴虚证型系数 = F2 and 肝气郁结证型系数 = A4 then TNM = H4

if 肝肾阴虚证型系数 = F3 then TNM = H4

if 肝肾阴虚证型系数 = F4 then TNM = H4

5 总结

在本次关于中医症素与乳腺癌 TNM 分期之间的关系的数据挖掘项目当中，我们可以发现，TNM 分期处于 IV 期的乳腺癌患者，其证型主要为：肝肾阴虚证、肝气郁结证以及脾胃虚弱证。其中肝肾阴虚证、肝气郁结证的临床表现较为突出，置信度与支持度都很高，且肝肾阴虚证几乎存在于所有的能得出 IV 分期的关联规则当中，因此，对于乳腺癌患者，应当选取以滋养肝肾的药物为主，并且，除了服用药物外，也可以在饮食上注意使用一些对提高肝功能和肾功能有帮助的食物。

肝脏是人体中的重要器官，乳腺癌患者的证型表示，其肝功能在一定程度上下降，因此除了关注癌细胞的扩散之外，也应时常对肝功进行检查，防止出现由乳腺癌诱发的并发症，加重病情。并且，肝气郁结表示患者的身心上已经不堪重负，心情以及生活态度都在发生转变，这就要求在治疗的过程中，注意对患者进行心理上的疏导，排除忧郁不安和焦虑的情绪，树立对抗病魔的决心。

6 小组分工明细

小组成员：李嘉伟，肖理想，尹吉宪，贺雨晴

李嘉伟：负责数据预处理部分的各项工作与算法实现，负责填写空值，数据离散化，主要算法实现为 K-means 聚类以及朴素贝叶斯分类器。

肖理想：负责对预处理的数据进行挖掘，得出有用的频繁项集，并根据频繁项集设置置信度来导出关联规则，主要算法为：Apriori 算法。

尹吉宪：负责对预处理的数据进行挖掘，导出频繁项集，在符合最小支持度的频繁项集上设置最小置信度来得出关联规则，主要算法：FP-growth 算法。

贺雨晴：负责对所得到的结果进行处理，寻找有用的关联规则，并通过分析得出必要的结论，然后通过决策树来得出各型与 TNM 分期之间的关系，主要算法：结果处理以及决策树的实现。

本次项目大家在积极完成各自的任务的前提下，帮助遇到困难的小组成员解决难题，在实践中对数据挖掘的各方面的知识进行了充分的利用，并通过此问题对乳腺癌也有了进一步的了解。