2016年深圳杯夏令营

D题参赛论文

基于系统生物学代谢综合征风险预测和干预的研究

参赛队员： 刘兴波（软件学院）

赵修竹（物理学院）

郭涵章（计算机科学与技术学院）

袁继伟（生命科学学院）

单位：山东大学

摘要

代谢综合征是一种以多种代谢性危险因素聚集为特征的临床症候群。各个致病因素复杂的相互作用，使得代谢综合征始终难以量化分析并对其发展做出合理预测和干预。近年来，代谢综合征在全球范围内的患病率在逐年上升，对代谢综合征患病风险的预测，并找寻致病的关键通路进而对个体实施个性化干预变得尤为重要。

本文先对基因数据进行预处理，去除无效基因，归一化，合并两批次数据。接着通过方差分析剔除无关基因，然后对所有个体进行聚类处理，得到不同的患病类别。在此基础上使用过滤式特征选择，分别采用 Relief-F 方法和条件互信息方法筛选相关基因，最后对每个基因的表达量高低与聚类得到的四种患病类型做独立性检验，从而得到与每种患病类型相关的致病基因。做出与筛选所得基因相关的代谢通路，预测个体患病程度时采用欧式距离进行相似性度量，用与待测个体最接近的样本预测其患病程度，最近样本所属类别的致病基因就是该个体的关键基因。

本文通过建立调控代谢综合征的网络模型，可以进一步了解代谢综合征的患病机理，并预测个体的患病程度以及风险，进而依据网络找到的关键通路对个体进行个性化的干预治疗，以降低代谢综合征的患病率。

关键词：代谢综合征；基因表达量；聚类分析；生物网络；特征选择；

目录

一、绪论

1.1 问题重述与分析

1.1.1 问题重述

利用题目所提供的样本基因表达数据，考虑包含基因组、表观基因组、转录组、蛋白质组、代谢组等多个组学之间的相互作用关系及外界影响，建立动态网络模型去预测待测个体罹患代谢综合征的风险或患病程度。结合基因通路信息、生物分子的相互作用关系，找到个体的关键致病通路，并据此提出个性化的干预治疗方案。

1.1.2 问题背景

代谢综合征[1]是以多种代谢性危险因素聚集为特征的代谢紊乱症候群，其代谢紊乱包括中心性肥胖、高血糖、高血压、血脂异常、高胰岛素血症等多种疾病，而这些代谢紊乱也是心、脑血管疾病及糖尿病的病理基础。代谢综合征可导致高血压、冠心病、心脑血管疾病、癌症等多种高死亡率的疾病的患病风险显著提高，极大影响人类健康。然而作为一类复杂的慢性疾病，代谢综合征的早期征兆不甚明显也使得对疾病的防治变得十分困难。

随着近年来生命科学高通量技术的发展，人类基因序列的信息已能够被解析，而人与人之间基因的差异也导致了不同人患有代谢综合征的风险有所不同。人体生理运行特征如基因表达量的定量测量使得对人体运行状态的动态监控成为可能。但同时，社会环境因素、生活方式、饮食习惯等外界因素也会对人体运行的状态产生或长期或短期的影响，最终导致的人体外部表型特征亦可以被测量量化。但人体是复杂的非线性系统，并非单一的基因信息抑或是表型特征即可对其加以描述，因此若想建立量化生命的动态模型，需要依托系统生物学的概念来进行，

系统生物学[1]旨在通过整合生命系统内不同性质的构成要素（基因、蛋白质、代谢小分子等）来对整个系统进行分析。

1.1.3 问题分析

针对 a 问，由于数据没有给出样本的患病类型，本文对所有个体聚类处理，然后采用过滤式特征选择的方法筛选与代谢综合征相关的基因，并对所有基因与个体类别进行独立性检验，找到与每个类别相关的基因，使用 Cytoscape 软件结合找到的基因制作出四种典型小分子物质的代谢调控网络。

针对 b 问，根据 a 问找到的基因，使用 Cytoscape 软件做出从每个基因出发到蛋白质合成，再到人体代谢的通路图，作为关键通路。

针对 c 问，根据相似性度量的原理，找到与待测个体欧式距离最接近的样本，预测待测个体的患病程度与该样本相同，并且对于待测个体来讲，其关键基因就是该样本在 a 问中聚类后所属类别的相关基因。

1.2 基本假设

本文不考虑多个基因只有同时表达异常才会导致患病的情况。

二、代谢综合征的系统生物学研究

2.1 系统生物学概述

系统生物学的概念是基于整体性和系统性的分析方法去阐释生物系统各个层次间相互作用关系及机理。整合性为系统生物学科的核心概念，一是体现在其学科交叉性，二则是体现在其对于系统组分研究方法上。系统生物学为一门交叉学科，整合了包括系统学、生命科学、信息科学、计算科学、数学等多种学科的知识体系。基于整体性的概念，系统生物学的理论基础为系统科学，信息科学、

数学、计算科学等为其提供方法论支持，而系统生物学的研究内容、实验数据等则由生命科学提供。而相比较于传统生物学一般只针对某个特定的生物分子进行研究，但无法依此对生物系统的特征以及调控模式进行描述，系统生物学整合了生物系统中包含的各种组分，并从机理上分析研究各组分之间的相互作用关系，从而能够更完善的阐释生物系统内的动态变化机制。

人体生命系统本质是复杂的多层次的生命系统，包含各种组分，如 DNA、氨基酸和各种代谢小分子等。基于系统科学的理论基础，系统生物学可研究这些组分间的相互作用关系，分析各种组分的不同特质及随空间、时间变化的模式。依据研究所得进行建模，来重现实验结果，分析系统的作用机制，同时可以通过模型的敏感性分析与鲁棒性分析，来发掘生物系统内部的一些关键调控节点，从而找寻一些组学间的关键通路。

2.2 组学分析方法概述

2.2.1 组学概念

组学本是属于分子生物学的概念，为一些种类个体的系统组合，如基因组学、转录组学、蛋白质组学、代谢组学，脂类组学，免疫组学和糖组学等。其中，基因组学、转录组学、蛋白质组学与代谢组学共同构成了系统生物学[2]的组学生物基础。

本文按照题意仅考虑基因组学、转录组学、蛋白质组学与代谢组学，因此在下文将仅对这四类组学进行详细介绍。

基因组学是研究生物基因组组成，组内各基因的精确结构、相互关系及表达调控机制的科学。应包括功能基因组学与结构基因组学两个主要方向。其中结构基因组学以全基因组测序为目标，而功能基因组学则是以鉴定基因功能为目标。

转录组学是研究整体水平上的基因转录情况及转录调控规律的科学。它旨在通过从 RNA 水平上探究基因表达的信息，并以此来推断未知基因的功能及特定基因间的调控规律。

蛋白质组学是在大分子层面上研究一个基因组或一个细胞、组织表达的全部

蛋白质的特征的科学，其包含蛋白质的表达水平、蛋白质被翻译后的修饰功能、蛋白质与蛋白质之间相互作用关系等方面的研究。旨在通过从蛋白质水平的了解来整体而全面的认识生物系统疾病发生、生理代谢等过程。

代谢组学是对某一生物或者细胞在某一特定生理状态或时刻下的所有低分子量的代谢产物同时进行定量与定性分析的一门科学。它通过仪器与分析技术相结合来监测生物机体在特定生理环境或生理状态下的代谢产物，并利用多元统计方法研究系统整体的生理功能情况。对代谢产物变化的分析，可以认识到基因水平的变化，因此对基因功能的探索可以通过对代谢组学的研究从微观的基因水平转换为宏观上的代谢分子水平。

2.2.2 组学分析方法

组学分析方法是一种依托多种组学数据来研究分析生物系统不同层次的动态变化特征的方法论。随着高通量科学技术在生命科学领域的发展，各种组学数据已能够被完善的测量收集。题目所提供的基因表达数据集也是组学数据中的一类，对其进行整合分析，来发掘生物系统内各个组分之间的内在调控关系，以描述整个生物体统的动态变化特征。通常对于特定基因、RNA 或某种蛋白质动态变化系统特征的研究，一般为给定输入信息，通过实验观察得到输出信息，但这样无法具体解释许多内在的作用机理和动力学特征，也就是说传统的方法会有黑箱的特质。而系统生物学的组学分析可以通过调控节点信息，进而由影响结果推断出具体节点影响的通路与功能，由此阐释生物系统内一些生理动态变化的过程及分子相互作用机理。

2.3 代谢综合征综述

2.3.1 代谢综合征定义及诊断标准

代谢综合征是一种以胰岛素抵抗为核心发病机制的代谢紊乱症候群。其临床表现主要为向心性肥胖、高血压、高胰岛素血症等多种代谢异常疾病。胰岛素抵抗不仅在在发展后期会导致病人的糖代谢恶化，进而发展为糖尿病，也与高血压、

高胰岛素血症、血脂异常增高等心血管疾病高危险因素相关。由此可见，代谢综合征患者会更易进一步发展患有糖尿病、心血管疾病等高死亡率的疾病。

近年来在全球医学领域达成共识的最新的代谢综合征诊断标准为 2009 年国际糖尿病联盟(IDF)、美国心脏协会（AHA)、美国国立卫生研究院（NIH)、美国心肺血研究所(NHLBI)联合发布的代谢综合征的诊断标准。在这一标准中：具备以下 5 项危险因素中 3 项及以上者定义为代谢综合征:

代谢综合征危险因素	测量判定标准
1.腹围升高	不同地区和种族人群具有不同标准
2.高密度脂蛋白胆固醇(HDL-C)降低：	HDL-C<40 mg/dl(1.03mmol/L)(男)，HDL-C<50 mg/dl(1.3 mmoL/L)(女)，或已确诊并治疗者
3.甘油三酯(triglycerides，TG)升高	TG≥150 mg/dl(1.7 mmol/L)，或已确诊并治疗者
4.血压升高	135/85 mmHg 或更高，或已确诊并治疗者
5.空腹血糖(FPG)升高	FPG≥100 mg/dl，或已确诊并治疗者

代谢综合征是一个不断完善发展的概念，这个概念的受到越来越多的关注很大程度上是由于近年来肥胖、二型糖尿病和心血管发病率的提升。因此代谢综合这一临床概念很大意义便在于对于二型糖尿病、心血管疾病发生进行预防。根据目前的流行病学资料，全世界大部分国家，有 20%一 30%的成年人患有代谢综合征，而且仍有不断增加的趋势。

2.3.2 代谢综合征研究概述

目前对于代谢综合症的研究多集中于，对代谢综合征疾病机理的研究，如代谢综合征的发病机制；对代谢综合征并发症的研究，如代谢综合征导致 II 型糖尿病、心血管疾病等疾病的风险；以及对代谢综合征相关基因[3]的研究，目前明确已知与代谢综合征相关的基因已有近几十种，如 LEP、INS、ADIPOQ、PPARG 等基因。对于代谢综合征的研究多集中在流行病学统计和临床数据分析两个方向。依靠临床数据的分析来判断患病程度是较为流行的诊断方法，但较难通过临床数

据来预测个体的患病风险或患病程度。而仅依靠高通量数据分析统计来建模预测，也会因丧失生物和医学机理的解释而难以准确预测。因此在近年的代谢综合征研究中，与代谢综合征相关的组学研究，如代谢组学、基因组学等，逐渐开始建立起来

2.3.3 代谢综合征与系统生物学

代谢综合征作为一个复杂的慢性疾病，已非传统生物医学的研究方法适应的领域，而系统生物学在分析阐释复杂生物系统的研究中体现出极大的优势，将系统生物学应用于代谢综合征的研究，可以得到完整而具体的结果.

系统生物学在关于代谢综合征的研究可以分为以下几个步骤：

1. 首先找到与代谢综合征相关的各个组分并对其深入了解和确定，在本文中，由于除基因表达数据外组学数据的缺失，仅重点考虑基因组学这一部分，描述基因相互作用网络，结合蛋白质网络、代谢网络等构造初步的系统模型。
2. 然后系统的改变网络模型中的部分节点或外部条件，观察在不同情况下系统组分的变化影响，即网络组分或结构发生的相应变化，整合相关信息，以找到一些关键的节点通路。
3. 最后将模型预测结果与文献中实验所得的结果进行比较，并不断对模型进行调试与改进，将模型完善并加以应用。

2.4 本文研究内容及意义

2.4.1 本文研究内容

本文根据给出的两个批次的患者基因表达量信息，利用方差、聚类、特征选择、独立性检验等方法筛选出与代谢综合征相关的基因，并做出这些基因相关的代谢通路，作为对患者病理学研究的依据，通过相似性度量来预测待测个体的患

病程度，并根据聚类结果找到患者的致病基因，以便今后给出有针对性的治疗方案。

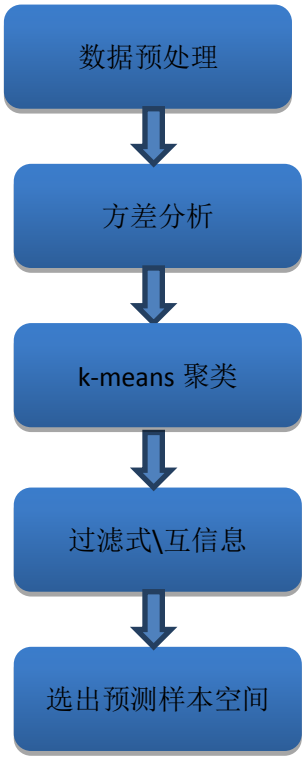
2.4.2 研究意义

若能根据题目所给数据建立模型，通过合理应用，就能够在人未患病的情况下提前预测其患病风险并及时实行预防措施，在较大程度上降低代谢综合征的患病率。

三、代谢综合征调控网络模型的构建

3.1 调控代谢综合征模型原理概述

根据代谢综合征的定义并结合题目数据，本文对题目数据进行预处理之后，采用方差分析做第一次筛选，然后通过 k-means 聚类，把剩余的样本分成若干类，然后用过滤式和互信息做第二次筛选，最后留下的样本及维度作为下一步患病程度预测的样本空间。



3.2 代谢综合征相关基因的筛选

3.2.1 基因表达数据预处理

数据的预处理是对进行主要处理前的一些方法，由于原始数据通常会有不完整或不一致的情况，大大降低了数据挖掘的效率，因此常采用数据清理、数据集成、数据变换等方法来对原始数据进行预处理修整，这个步骤对数据挖掘有着十分重要的意义。

数据清理：为了便于后续的处理和减小基因突变的影响，本文首先去掉了样本数据中基因表达量为零个数超过 90 的基因，然后出于监督的目的，又去掉了未知基因，去掉了未知患病程度的样本。

数据集成：样本数据中 0 批次和 1 批次除了基因数目不同外没有其他区别，为了增加样本数量，本文取 0 批次和 1 批次基因的交集，取 0 批次和 1 批次样本个体的并集。

数据变换：将基因表达量做归一化[4]处理。归一化的处理函数：

$$x^* = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \qquad (1)$$

3.2.2 基因表达数据冗余信息的剔除

由于本题给出的数据都是患病的，无法根据患病和不患病的基因表达量的差异显著性来寻找致病基因。此外，由于维数远高于样本数量，带来了维数灾难，寻常的回归分析、因子分析、判别分析、通径分析不再适用。本文首先采用统计学分析。方差分析是一种简单的统计学分析方法，它可以用来衡量多组数据之间的差异性。本文根据四种患病程度基因表达量的方差分析来初步筛选出有显著性差异的基因。计算所有组基因表达量的方差[5]，它们的频率分布直方图如图 1 所示：

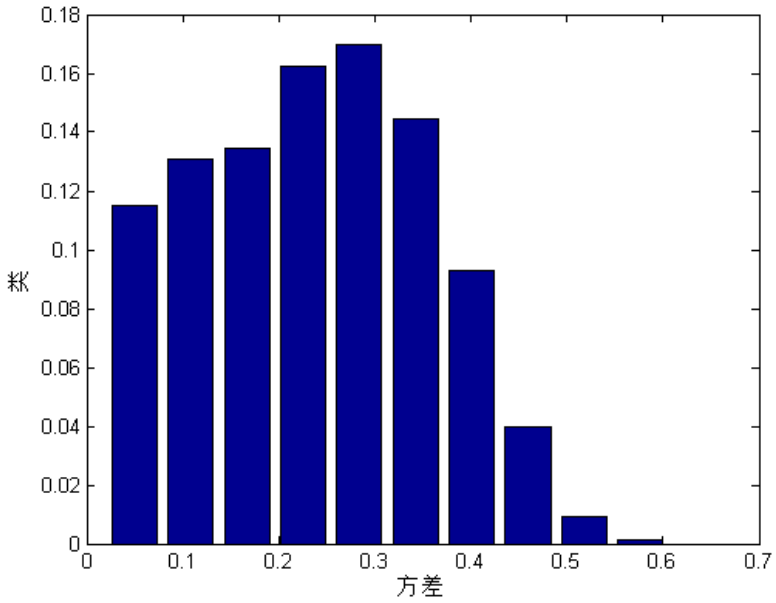


图 1　基因表达量方差直方图

通过基因表达量直方图可以看出，大部分基因方差较小，表现出较小的差异性；方差大于 0.3 后，频率急剧下降，这部分基因表现出较大的差异性，即本文需要的基因。根据频率分布，我们选取方差最大的 5000 个基因做后续研究。

3.2.3 基因表达数据聚类分析

基于方差分析初步筛选剩下的 5000 个基因，考虑到代谢综合征是一系列病的综合体现，我们对样本进行聚类。

$k-means$[6]是经典的聚类算法之一。k-means 算法快速、简单,对大数据集处理有较高的效率,且时间复杂度近于线性，而且适合挖掘大规模数据集。本文使用 k-means 算法，以 k 为参数，把方差分析筛选出来的 5000 个基因分成 k 个类，使类内具有较高的相似度，而簇间的相似度较低，即类间分离度较高，类内聚集度较好。k-means 算法的处理过程如下：首先，随机地选择 k 个对象，每个对象初始地代表了一个簇的平均值或中心;这种随机性会带来聚类上的不确定性，即每次聚类结果都会有细微差别。对剩余的每个基因，根据其与各类中心的距离，将它赋给最近的类;然后重新计算每个类的平均值。不断重复这个过程，直到准则函数收敛。通常，采用平方误差准则，其定义如下：

$$E=\sum_{i=1}^{k}\sum_{p\in C_i}\left|p-m_i\right|^2 \qquad (2)$$

此处E是数据库中所有对象的平方误差的总和，p是空间中的点，m_i是簇Ci的平均值。该目标函数使生成的簇尽可能紧凑独立

k-means 聚类算法的算法流程如下：

输入：包含 n 个对象的数据库和簇的数目 k；

输出：k 个簇，使平方误差准则最小。

步骤：

(1) 任意选择 k 个对象作为初始的簇中心；

(2) 迭代；

(3) 根据簇中对象的平均值，将每个对象重新赋予最类似的簇；

(4) 更新簇的平均值，即计算每个簇中对象的平均值；

(5) 直到对象所属的簇不再发生变化。

基于 AIC 准则或者 BIC 准则可以给出最佳的聚类数目，但由于本文待聚类的数据维数高于样本数目，无法采用 AIC 准则或者 BIC 准则，所以采用聚类后可视化剪影值的方法来寻找最佳聚类数目。分剪影值是衡量聚类好坏的一个数值，值越大表示，聚类效果越好。

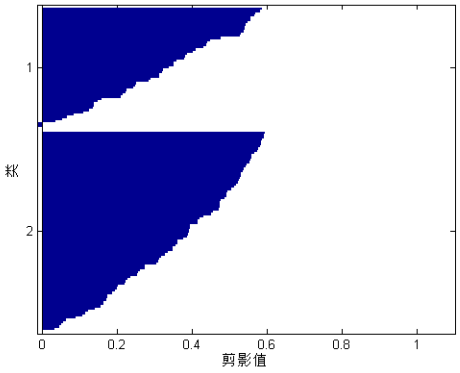


图 2　2-means

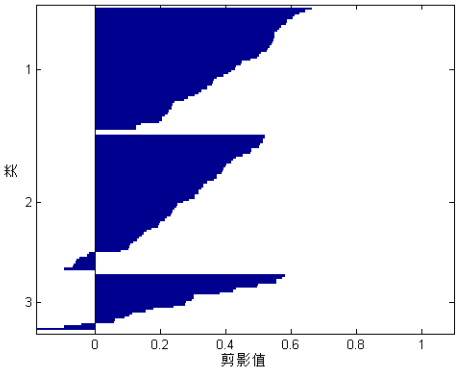


图 3　3-means

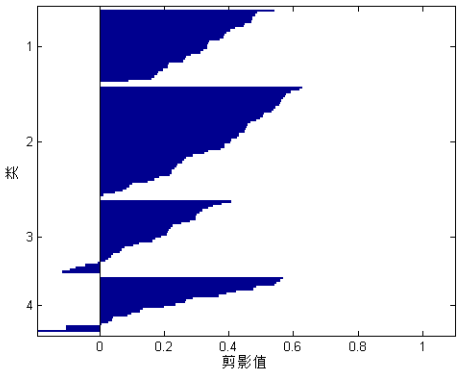


图 4　4-means

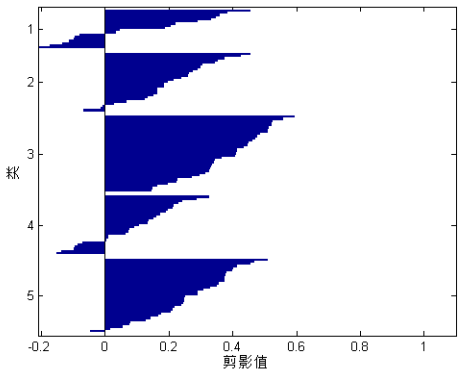


图 5　5-means

根据聚类的图示效果和问题实际情况，本文的 k-means 的 k 值取 4。

3.2.4 基于过滤式的基因筛选

3.2.4.1 特征选择概述

特征选择也叫特征子集选择，是从原始特征中选择出一些最有效特征以降低数据集维度的过程，是在高维、小样本基因表达数据上进行有监督学习的必要预处理。本文剩余的 5000 个基因中存在着大量与代谢综合征不相关的基因，由于本文之后的相似性度量采用了欧式距离作为评判标准，这些不相关基因会打乱欧式空间，对寻找相似个体产生巨大的干扰，例如原本待测个体与数据集中个体 A 在相关基因方面最相似，但在某些与疾病不相关基因方面存在较大差异，这样就会导致相似个体的寻找结果不理想。本文只留下对区分个体有较大贡献的基因，提高了之后相似性度量的精度，降低了算法的复杂度。

特征选择的一般分为两个环节，第一个环节是子集搜索，第二个环节是子集评价。给定一个大小为 d 的特征集合 S，我们可以将每个特征看作一个候选子集，对这 d 个单特征候选子集进行评价，选择最优的集合作为第一轮的选定集；然后在上一轮的选定集中加入一个特征，构成包含两个特征的候选子集，对这 d-1 个候选两特征子集进行评价，再次选择最优的集合作为本轮的选定集；假定在 k+1 轮时，最优的候选（k+1）特征子集不如上一轮的选定集，则停止产生候选子集，并将上一轮选定的 k 特征集合作为特征选择的结果。这样逐渐增加相关特

征的策略称为这样逐渐增加相关特征的策略称为“前向”搜索。本文将要介绍的两种筛选方式都采用了前向搜索的策略。

对于子集评价问题，特征子集 A 实际上确定了对数据集 D 的一个划分，而样本标记信息 Y 则对应着对 D 的真实划分，通过估算这两个划分的差异，就能对 A 进行评价.与 Y 对应的划分的差异越小，则说明 A 越好。相关系数、t-检验的 p 值、信息熵等能判断两个划分差异的机制都能用于特征子集评价。

本文将基于过滤式的特征选择方法对代谢综合症相关基因进行筛选。过滤式特征选择算法一般针对每个特征计算一个特定的统计值，再根据这个统计值的大小进行特征选择。主要用到的统计量有相关系数、t-检验的 p 值，Fisher 比例互信息等。后文将分别采用 Relief-F 与互信息对上文的 5000 个基因进行过滤，并比较两种筛选方法的优缺点。

3.2.4.2 Relief-F 方法

$Relief$[7]的“相关统计量”是一个向量，在本文中其每个分量对应一个基因，而基因子集的重要性则是由子集中每个基因所对应的相关统计量分量之和来决定。本文指定欲选取的基因个数 k，然后选择相关统计量分量最大的 k 个基因。对每个样本x_i，Relief 先在x_i的同类样本中寻找其最近邻$x_{i,nh}$，称为“猜中近邻”，再从x_i的异类样本中寻找其最近邻$x_{i,nm}$，称为“猜错近邻”，然后，相关统计量对应于基因j的分量为

$$\delta^j = \sum_i -diff\left(x_i^j, x_{i,nh}^j\right)^2 + diff\left(x_i^j, x_{i,nm}^j\right)^2 \qquad (3)$$

其中 x_a^j 为个体 x_a 在基因 j 上的取值，基因的表达量为连续型，所以取 $diff\left(x_a^j, x_b^j\right)=\left|x_a^j - x_b^j\right|$ ， x_a^j, x_b^j 已归一化到[0,1]区间。

从公式（3）可看出，若个体x_i与其猜中近邻$x_{i,nh}$在基因j上的距离小于x_i与其猜错近邻$x_{i,nh}$的距离，则说明基因j对区分同类与异类样本是有益的，于是增大属性j所对应的统计量分量；反之，则说明基因j起负面作用，于是减小基因j所

对应的统计量分量。最后，对所有个体每个基因的统计分量加和，就得到各基因的相关统计量分量，分量值越大，说明该基因对分类起到的贡献越大。

前文将样本集分为了四类，这里采用扩展变体 Relief-F 处理多分类问题。对样本x_i，若它属于第 k 类，则 Relief-F 先在第 k 类的样本中寻找x_i的最近邻$x_{i,nh}$并将其作为猜中近邻，然后在第 k 类之外的每个类中找到一个x_i的最近邻作为猜错近邻，记为 $x_{i,\mathrm{nh}}$ $(l = 1,2,3,4; l! = k)$。于是，相关统计量对应于基因j的分量为

$$\delta^j = \sum_i -diff\left(x_i^j, x_{i,nh}^j\right)^2 + \sum_{l\neq k}\left(pl \times diff\left(x_i^j, x_{i,l,nm}^j\right)^2\right) \qquad (4)$$

其中pl为第l类样本在样本集中所占的比例。

该算法流程如下：

1. 计算每个基因对应的相关统计量的分量，作为该基因对分类的贡献。

2. 对基因按贡献大小排序。

3. 选择贡献最大的 k 个基因作为结果基因集。

本文采用该方法对之前处理出的 5000 个基因进行了筛选，使用了 0 和 1 批次共 141 个个体，取统计量阈值为 0，即只保留对分类贡献为正的基因，取得了较好的效果，第一次过滤剩余 1028 个基因，第二次过滤剩余 783 个基因，第三次过滤剩余 728 个基因，之后基因数量逐渐趋于稳定，于是我们取第二次剩余的 783 个基因作为筛选的结果。每次筛选的算法复杂度为 O(5000*141*141)，运行效率极高。

Relief-F 方法相比互信息筛选得到的基因会存在关系上的冗余，但对基因贡献的计算使得筛选不易受到个别个体的干扰，相比互信息也存在着优势，这里会放在后文进行说明。

3.2.4.3 条件互信息方法

在信息论中，互信息[8]用来度量变量间的相关信息。与相关系数等统计量相比，互信息对噪声鲁棒，能够获取变量间的非线性关系。

信息熵用来度量随机变量的不确定程度。对基因表达量f，其熵$H(f)$的度量

如公式（5）所示，$p(f)$为概率密度函数，$R(f)$为f的定义域。

$$H(f)=-\int_{R(f)} p(f)\log(p(f)) \tag{5}$$

对于个体的样本类别，令$p(y)$为y的概率分布函数，C 为所有类别的集合，则其熵的定义如公式（6）所示。

$$H(y)=-\sum_{y\in C} p(y)\log(p(f)) \tag{6}$$

基因表达量f和样本类别y互信息的计算公式如（5）所示

$$I(y;f)=H(f)+H(y)-H(f,y) \tag{7}$$

基因选择就是希望寻找尽可能小的基因集合，去消除尽可能多样本类别的不确定程度。本文选取 k 个与样本类别具有最大互信息的基因，并去除基因之间信息的冗余。

为解决基因的冗余性问题，令S_k表示已经选择了 k 个基因的集合，$MI(y;f\mid S_k)$为$I(y;f\mid S_k)$的近似度量，则该近似度量如公式（8）所示。

$$MI(y;f\mid S_n)=\min_{f'\in S_n} I(y;f\mid f') \tag{8}$$

本文采用 MIGS 特征选择算法进行子集搜索。算法流程如下：

1. 初始化，S_0被为空，每个基因的条件互信则被初始化为该基因表达量和样本类别的互信息。

$$MI(y;f_j\mid S_0)=MI(y;f_j), j=1,2,...,m \tag{9}$$

2. 每次顺序扫描候选基因集 G，找出其中最具有最大$MI(y;f\mid S_{k-1})$的基因，并把它加入到基因集中。该选择过程由公式（9）所示:

$$S_k=S_{k-1}\cup \arg\max_{f_j\in G}\{MI(y;f_j\mid S_{k-1})\} \tag{10}$$

3. 根据最新的基因集S_k更新相关基因的条件互信息，更新规则如下：

$$MI(y;f_j\mid S_k)=\begin{cases}\min\{MI(y;f_j\mid S_{k-1}),MI(y;f_j\mid S_k-S_{k-1})\} & f_j\notin S_k\\ 0 & f_j\notin S_k\end{cases} \tag{11}$$

上述算法需要根据样本精确估计 $H(y)$、$H(y\mid f)$和$H(y\mid f,f')$，下文取 $H(y\mid X)$ 作为它们的一般形式。由于可以所使用的样本数量较少，本文采用核密度估计方法。

已知 $H(y|X)$ 的计算如公式（12）所示:

$$H(y|X)=-\int_{x,y} p(y,X)\log(p(y|X))dXdy \tag{12}$$

由于样本类别 y 是离散值，因此 $H(y|X)$ 可以表示为公式:

$$H(y|X)=-\int_{X} p(X)\sum_{k=1}^{l} p(c_k|X)\log(p(c_k|X))dX \tag{13}$$

根据贝叶斯公式， $p(c_k|X)$ 的估计如公式（14）所示:

$$p(c_k|X)=\frac{p(X,c_k)}{\sum_{i=1}^{l}p(X,c_j)}=\frac{p(X|c_k)p(c_k)}{\sum_{i=1}^{l}p(X|c_j)p(c_j)} \tag{14}$$

其中 $p(c_k)=\frac{n_{c_k}}{n}$，n_{c_k} 为 C_k 类样本的数目，n为所有样本的数目。$p(X|c_k)$ 则可以采用如（15）所示的核密度估计算法获得。

$$p(X|c_k)=\frac{1}{n_{c_k}}\sum_{y_i=c_k}K(X-X_i,h)=\frac{1}{n_{c_k}}\sum_{y_i=c_k}\frac{1}{\left(\sqrt{2\pi}h\right)^m}\exp\left(-\frac{(X-X_i)^T(X-X_i)}{2h^2}\right) \tag{15}$$

其中，$K(X-X_i,h)$是核函数，本文采用高斯核函数，参数h使用基于最小化二次误差积分的最优核函数参数，取 $h=\left(\frac{4}{(2m+1)n}\right)^{\frac{1}{m+4}}$，其中$m$为为样本的维度，$n$为样本的数量。

结合公式（14）（15），可得 $p(c_k|X)$ 的估计值如公式（16）所示。

$$p(c_k|X)=\frac{\frac{1}{n_{c_k}}\sum_{y_i=c_k}\exp(-\frac{(X-X_i)^T(X-X_i)}{2h^2})}{\sum_{j=1}^{l}\frac{1}{n_{c_k}}\sum_{y_i=c_k}\exp(-\frac{(X-X_i)^T(X-X_i)}{2h^2})} \tag{16}$$

然后，利用数值积分计算 $H(y|X)$ 。

本文采用该方法对前面的到的 5000 个基因进行筛选，保留了前 50 个基因，于是该算法复杂度为 0(5000*500+5000*141)，其中积分的计算降低了算法的效率，运行时间略长于前面的 Relief-F。

该方法充分利用了基因之间的信息包含关系，相比 Relief-F 去除了存在冗余的基因。但是，个别个体对信息熵的计算可能产生较大影响，例如，有一个患者基因 A 的表达量较高，而又恰好被分在类 p 中，而其他患者该基因表达正常，由于样本数量较少，信息熵会认为该基因降低了类 p 的不确定程度，于是该基因被保留，而实际上基因 A 可能只是一个与代谢综合征无关的基因。相比之下，Relief-F 则避免了这种局面的出现，因为若同类患者选择该患者为最近邻，则计算该基因统计分量为正，不同类患者则为负，而不同类患者数量大于同类患者，于是该基因若与代谢综合征无关，则最终统计量的计算结果将为负值。

本文利用 GeneCards 数据库的文献挖掘功能搜集了 5000 个可能与代谢综合征相关的基因，与前面两种方法得到的基因进行对比，结果如下：利用 Relief-F 方法得到的 783 个基因中，排在前 5000 的有 349 个，排在前 2000 的有 134 个，前 500 的有 42 个，前 50 的有 11 个；利用条件互信息方法得到的 50 个基因中，前 5000 有 32 个，前 2000 有 14 个，前 500 有 6 个，前 50 有 1 个。这里由于互信息方法继续选取基因的难度会加大，命中率会呈现下降趋势，所以本文认为 Relief-F 方法命中率更高，所以之后的研究将基于这 783 个基因进行。

3.2.5 基于独立性检验的基因分类

本文将前面筛选到的每个基因的表达量高低与聚类得到的四种患病类型做独立性检验[9]，来区分与每个患病类型相关的基因。

这里我们选取每个基因表达量的中位数作为阈值，检验方式选择卡方检验，可信程度取 99%，得到 300 个与类型一相关的基因，446 个与类型二相关的基因，375 个与类型三相关的基因，107 个与类型四相关的基因。

一个基因不止与一种患病类型相关，而独立性检验成功地将每个基因分到了多个类型中。

3.3 基于 Cytoscape 软件的代谢关键通路构建

本文从筛选的基因中挑选了 30 个与代谢综合征密切相关的基因，构建了它们调控的代谢通路，如下图所示，

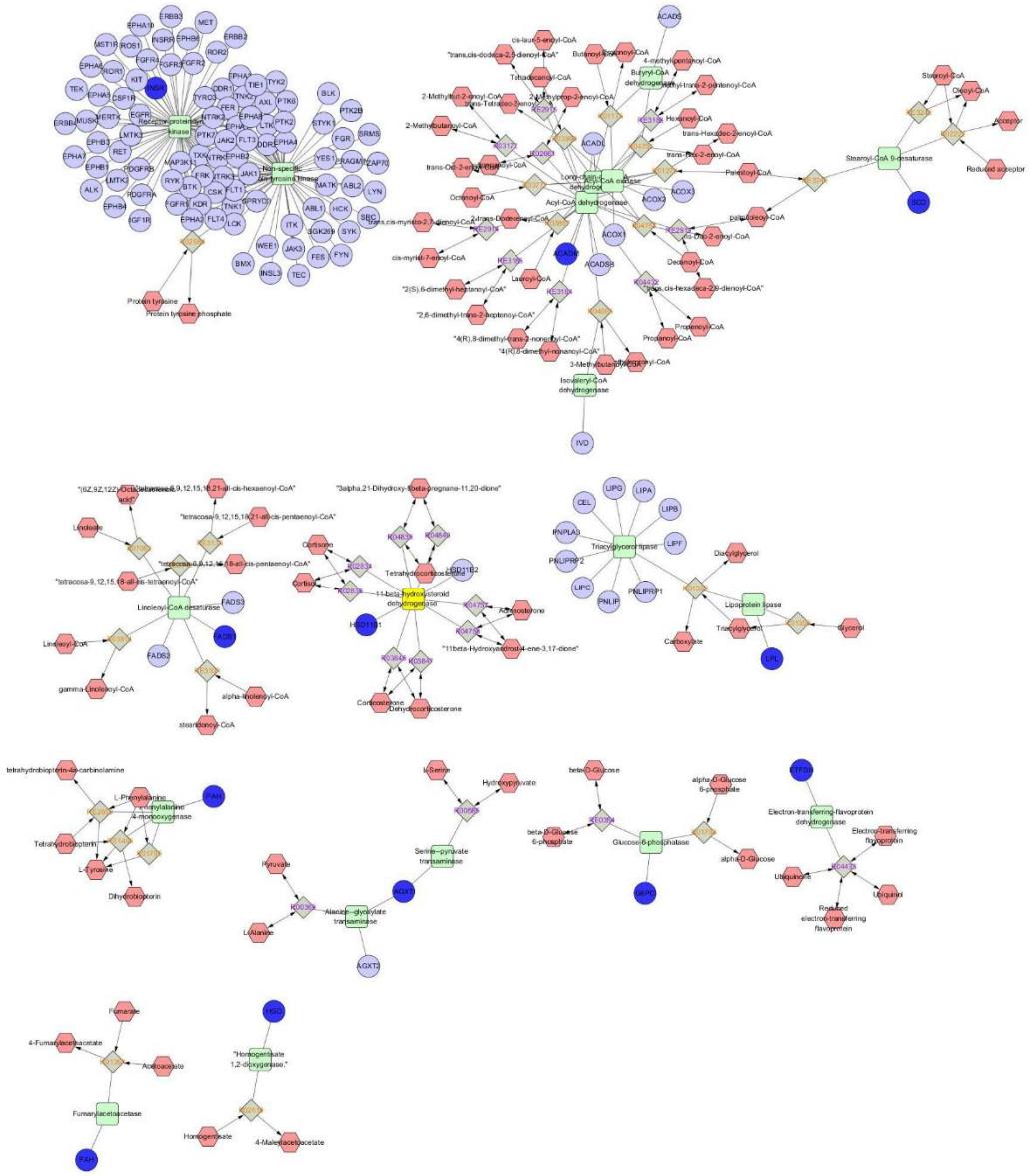


图 6 代谢关键通路图

3.4 于 Cytoscape 软件的调控网络构建

本文基于 Cytoscape 软件从物质的角度，结合前面找到的基因，制作了葡萄糖、胆固醇、脂蛋白、三酰甘油四种典型小分子物质的调控网络，该网络给出了各组学之间的关系，给出具体数据即可根据此网络构建贝叶斯网络来预测个体的患病风险。调控网络如下图所示。

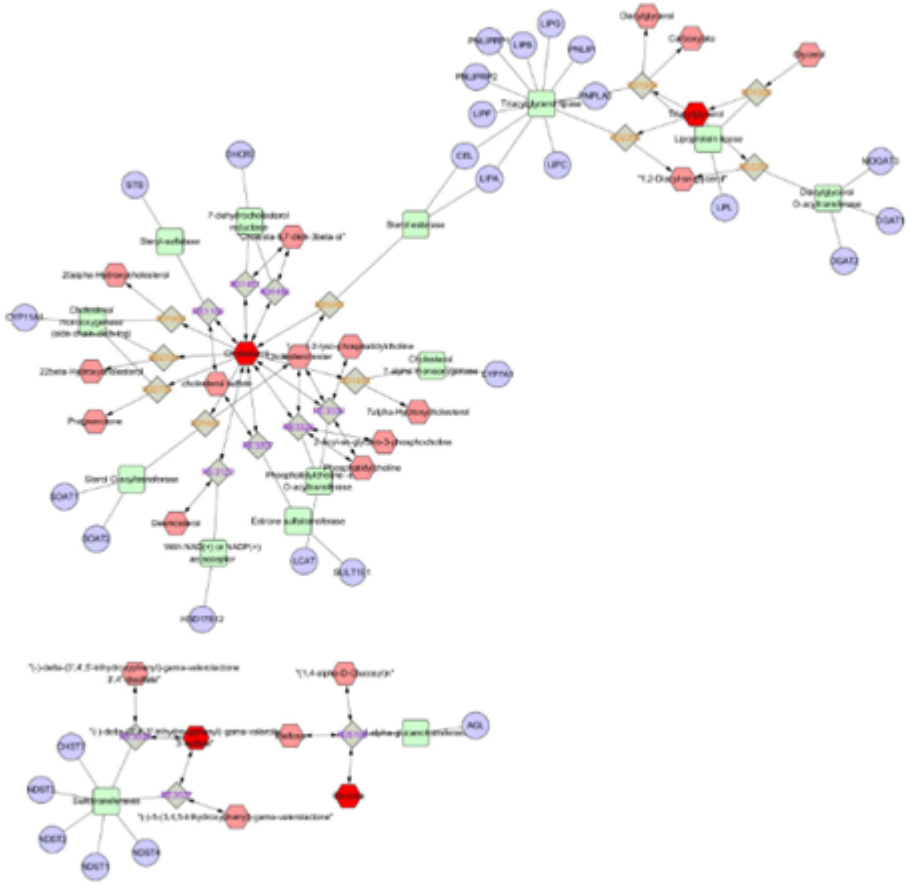


图 6 代谢调控网络图

四、代谢综合征患病程度预测模型的构建

4.1 患病程度预测模型原理概述

预测可以采用分类算法，如 SVM、神经网络、贝叶斯学习等机器学习的方法，但是本文多次筛选过滤之后剩下的基因数目仍然多于样本数目，所以本文采取了基于相似性度量的预测，这是一种简单快速且有效的方法。

基于题目给出的数据，假设有一未知患病程度的基因表达量数据的样本，本文通过相似性度量来寻找最相似的样本。

样本为a,测试数据为$b = (x_1, x_2, \dots, x_n)$。

不同样本之间的相似性度量，通常采用的方法就是计算样本间的“距离”。本文通过寻找最小的距离来寻找最相似的样本。

4.2 基于相似性度量的患病程度预测模型

最小的距离的模型如下:

$$s.t.\ min\ d \tag{17}$$

欧氏距离是最易于理解的一种距离计算方法，源自于欧氏空间中两点间的距离公式，本文采用欧氏距离来衡量，即：

$$d = \sqrt{\sum_{k=1}^{n} \left(x_1 - x_{k1}\right)^2} \tag{18}$$

与测试数据最相似的样本所属的患病类型即要预测的患病类型。如果样本数目足够大而且具有代表性，基于相似性度量的方法可以得到较为准确的预测正确率。

本文先实验了采用距离待测个体最近的一个样本预测其患病程度，每次从 141 个样本中选取 10 个作为待测个体，在其余 131 个样本中寻找最近邻，交叉验证 5 次，每次预测正确的个体数量分别为 3，4，2，3，4，平均命中率为 32%，只比随机猜测高 7 个百分点。

为解决上述问题，本文对方案进行了调整，每次选取距离待测个体最近的 5 个样本，取出现次数最多的患病程度作为预测的患病程度，若出现同时有两个患

病程度有两个样本，则取两个样本欧式距离之和更近的患病程度作为预测结果，这样重做之后预测的平均命中率提高到 38%。

五、本文研究的优缺点分析

5.1 本文研究的优点

1.科学性:本文研究时充分考虑到了组学的含义，也考虑到了基因表达量在统计学上的分布特性。

2.创新性：本文研究时采用了层层筛选的方法，每一层的筛选合理且有效。

3.普适性：本文研究所使用的方法适用于同类问题的研究，对于高维少样本的特征提取具有推广价值。

5.2 本文研究的缺点

1.多个基因可能只有在同时变异的情况下才会致病，本文的忽略了这种情况，只考虑单基因影响，多基因情况可能对基因的筛选产生影响。

2.本文只研究了部分与代谢综合征相关的基因，待测个体可能存在本文没有研究到的致病基因。

六、研究展望

本文在研究上由于数据和时间的限制，存在一些可以继续深入研究的地方。如果能够有人体六个组学的数据则可以建立调控网络，如果样本数量足够大的话，一些常规的分类算法、特征提取算法都可以使用，这对致病基因的筛选、关键通路的研究都有重要意义。

本文所提出的层层过滤筛选的方法对于同类问题具有普适性，但如果样本数据代表性不够强，会带来较大误差，如果能根据文献资料人工的筛选掉一些干扰基因，对患病程度的预测的准确性会有很大的提升。

七、参考文献

[1] 赵峥.代谢综合征的研究进展[J]. 实用心脑肺血管病杂志,2011,01:143-144.

[2]齐云峰. 基于系统生物学方法的干扰素-γ 和白介素-6 信号转导通路建模以及抗癌药物诱导细胞凋亡机制的研究[D].东北师范大学,2014.

[3]杨丽兰,华琦. 代谢综合征的基因多态性研究进展[J]. 医学研究生学报,2008,05:553-556.

[4] 肖汉光，蔡从中.特征向量的归一化比较性研究[J].计算机工程应用,2009,45(22):117-119.

[5] 杨小勇.方差分析法浅析——单因素的方差分析[J].实验科学与技术,2013,11(1):41-43.

[6]袁方，周志勇，宋鑫.初始聚类中心优化的 k-means 算法[J].计算机工

程,2007,33(03):65-66.

[7] 周志华.机器学习[M].清华大学出版社，2016：247-250.

[8] 蔡瑞初.基因表达数据挖掘若干关键技术研究[D]:华南理工大学，2010：25-32.

[9]司守奎，孙玉箐，数学建模算法与应用[M]:国防工业出版社,193-261,208-211,2011.08.

八、附录

1.方差计算:

```
import java.io.*;
import java.util.*;

public class Test5 {
    static int n,m,ct;
    static double u,s2,rr;
    static int[] cnt=new int[5];
    static int[] stage=new int[105];
    static String sp;
    static String[] name=new String[20105];
    static double[] sx=new double[20105];
    static double[] ss=new double[20105];
    static double[][] bb=new double[20105][105];
    static double[][] aa=new double[20105][105];
    static Pair[] pp=new Pair[20105];
    static String[] str=new String[20105];
    static void normal(int k)
    {
```

```
        double max=0.0,min=100000000.0;
        for(int i=1;i<=m;i++)
        {
            max=Math.max(max,bb[k][i]);
            min=Math.min(min,bb[k][i]);
        }
        for(int i=1;i<=m;i++)
            aa[k][i]=(bb[k][i]-min)/(max-min);
    }
    static class Pair implements Comparable<Pair>
    {
        double x;int y;
        Pair(double a,int b)
        {
            x=a;y=b;
        }
        public int compareTo(Pair p) {
            if(x>p.x) return -1;
            if(x<p.x) return 1;
            return 0;
        }
    }

    public static void main(String[] args) throws IOException {
        FileWriter fw=new FileWriter(new
File("D:/data/ex0_ss.txt"),false);
        BufferedWriter ww=new BufferedWriter(fw);
        Scanner in=new Scanner(new File("D:/data/ex0_2.txt"));
        String s=in.nextLine();
        Scanner sc=new Scanner(s);
        while(sc.hasNext())
            stage[++m]=sc.nextInt();
        System.out.print(m);
        while(in.hasNext())
        {
            s=in.nextLine();
            sc=new Scanner(s);
            name[++n]=sc.next();
            str[n]=s;
            for(int i=1;i<=m;i++)
                bb[n][i]=sc.nextDouble();
            normal(n);
        }
        System.out.println(n);
```

```
        for(int i=1;i<=n;i++)
        {
            for(int j=1;j<=m;j++)
                ss[i]+=aa[i][j];
            ss[i]/=m;
            for(int j=1;j<=m;j++)
            {
                rr=aa[i][j]-ss[i];
                sx[i]+=rr*rr;
            }
            sx[i]/=m;
        }
        for(int i=1;i<=n;i++)
            pp[i]=new Pair(ss[i],i);
        Arrays.sort(pp,1,n+1);
        for(int i=1;i<=n;i++)
        {
            ww.write(""+ss[i]);
            ww.newLine();
        }
        ww.close();
        System.out.println(ct);
    }
}
```

2. 方差直方图：

```
A=importdata('D:/data/ex0_ss(4).txt');
[a,b]=hist(A);
bar(b,a/sum(a))
```

3. k-means 聚类：

```
X=importdata('D:/data/ex_ss.txt');
[cidx2,cmeans2,sumd2,D2]  =  kmeans(X',5,'dist','sqEuclidean');
P2  =  figure;clf;
[silh2,h2]  =  silhouette(X',cidx2,'sqeuclidean');
ylabel('类');
xlabel('剪影值');
```

4.互信息计算:

```
import java.io.*;
import java.util.*;

public class Test3 {
    static int n,m,p;
```

```
static double r,hy,h;
static int[] cnt=new int[5];
static double[] pp=new double[5];
static int[] mm=new int[20105];
static int[] stage=new int[105];
static String[] name=new String[20105];
static double[] mi=new double[20105];
static double[] hh1=new double[20105];
static double[] hh2=new double[20105];
static double[][] bb=new double[20105][105];
static double[][] aa=new double[20105][105];
static void normal(int k)
{
    double max=0.0,min=100000000.0;
    for(int i=1;i<=m;i++)
    {
        max=Math.max(max,bb[k][i]);
        min=Math.min(min,bb[k][i]);
    }
    for(int i=1;i<=m;i++)
        aa[k][i]=(bb[k][i]-min)/(max-min);
}
static double geti1(int a)
{
    return hy-hh1[a];
}
static double geti2(int a,int b)
{
    return hh1[b]-hh2[a];
}
static double geth1(int a)
{
    double res=0.0,s,ss,rr;
    for(int i=0;i<20;i++)
    {
        s=ss=0.0;
        double x=i*0.05+0.025;
        for(int k=1;k<=4;k++)
            pp[k]=getpp(x,k,a);
        for(int k=1;k<=4;k++)
        {
            rr=getp(x,k);
            ss+=rr*Math.log(rr);
            s+=pp[k]*cnt[k];
```

```
        }
        s/=m;
        res+=s*ss;
    }
    res*=-0.05;
    return res;
}
static double geth2(int a,int b)
{
    double res=0.0,s,ss,rr;
    for(int i=0;i<20;i++)
    for(int j=0;j<20;j++)
    {
        double x=i*0.05+0.025;
        double y=j*0.05+0.025;
        s=ss=0.0;
        for(int k=1;k<=4;k++)
            pp[k]=getpp2(x,y,k,a,b);
        for(int k=1;k<=4;k++)
        {
            rr=getp(x,k);
            ss+=rr*Math.log(rr);
            s+=pp[k]*cnt[k];
        }
        s/=m;
        res+=s*ss;
    }
    res*=-0.0025;
    return res;
}
static double getpp(double x,int k,int a)
{
    double res=0.0,xx;
    h=4.0/3.0/m;
    h=Math.pow(h,0.2);
    for(int i=1;i<=m;i++)
    {
        if(stage[i]!=k) continue;
        xx=Math.abs(x-aa[a][i]);
        res+=Math.exp(-xx*xx/2.0/h/h);
    }
    res/=cnt[k]*Math.sqrt(2.0*Math.PI)*h;
    return res;
}
```

```
    static double getp(double x,int k)
    {
        double res=0.0,rr,qq=0.0;
        for(int i=1;i<=4;i++)
        {
            rr=pp[i]*cnt[i]/m;
            if(i==k) qq=rr;
            res+=rr;
        }
        res=qq/res;
        return res;
    }
    static double getpp2(double x,double y,int k,int a,int b)
    {
        double res=0.0,xx,yy;
        h=4.0/5.0/m;
        h=Math.pow(h,1.0/6.0);
        for(int i=1;i<=m;i++)
        {
            if(stage[i]!=k) continue;
            xx=Math.abs(x-aa[a][i]);yy=Math.abs(y-aa[b][i]);
            res+=Math.exp(-(xx*xx+yy*yy)/2.0/h/h);
        }
        res/=cnt[k]*2.0*Math.PI*h*h;
        return res;
    }

    public static void main(String[] args) throws IOException {
        FileWriter fw=new FileWriter(new
File("D:/data/test0_2.txt"),false);
        BufferedWriter ww=new BufferedWriter(fw);
        Scanner in=new Scanner(new File("D:/data/ex0_3.txt"));
        String s=in.nextLine();
        Scanner sc=new Scanner(s);
        while(sc.hasNext())
            stage[++m]=sc.nextInt();
        while(in.hasNext())
        {
            s=in.nextLine();
            sc=new Scanner(s);
            name[++n]=sc.next();
            for(int i=1;i<=m;i++)
                bb[n][i]=sc.nextDouble();
            normal(n);
```

```
        }
        for(int i=1;i<=m;i++)
            cnt[stage[i]]++;
        for(int i=1;i<=4;i++)
            hy+=((double)cnt[i]/m)*Math.log((double)cnt[i]/m);
        hy=-hy;
        for(int i=1;i<=n;i++)
            hh1[i]=geth1(i);
        for(int i=1;i<=n;i++)
            mi[i]=geti1(i);
        for(int o=1;o<=50;o++)
        {
            r=0.0;
            for(int i=1;i<=n;i++)
                if(mi[i]>r) { r=mi[i];p=i;}
            mm[p]=1;
            ww.write(name[p]);
            ww.newLine();
            System.out.println(name[p]);
            for(int i=1;i<=n;i++)
                if(mm[i]==0) hh2[i]=geth2(i,p);
            for(int i=1;i<=n;i++)
            {
                if(mm[i]==1) mi[i]=0.0;
                else mi[i]=Math.min(mi[i],geti2(i,p));
            }
        }
        ww.close();
    }
}
```

5.Relief-F:

```
import java.io.*;
import java.util.*;

public class Test6 {
    static int n,m,p;
    static double hh,ll,eps=1e-9;
    static int[] cnt=new int[5];
    static double[] rr=new double[5];
    static int[][] mm=new int[205][5];
    static int[] stage=new int[205];
    static String[] name=new String[20105];
    static double[][] dis=new double[205][205];
    static double[][] bb=new double[20105][205];
```

```
    static double[][] aa=new double[20105][205];
    static String[] str=new String[20105];
    static Pair[] pp=new Pair[20105];
    static class Pair implements Comparable<Pair>
    {
        double x;int y;
        Pair(double a,int b)
        {
            x=a;y=b;
        }
        public int compareTo(Pair p) {
            if(x>p.x) return -1;
            if(x<p.x) return 1;
            return 0;
        }
    }
    static void normal(int k)
    {
        double max=0.0,min=100000000.0;
        for(int i=1;i<=m;i++)
        {
            max=Math.max(max,bb[k][i]);
            min=Math.min(min,bb[k][i]);
        }
        if(min==max) return;
        for(int i=1;i<=m;i++)
            aa[k][i]=(bb[k][i]-min)/(max-min);
    }

    public static void main(String[] args) throws IOException {
        FileWriter fw=new FileWriter(new
File("D:/data/get_ss3.txt"),false);
        BufferedWriter ww=new BufferedWriter(fw);
        Scanner in=new Scanner(new File("D:/data/get_ss2.txt"));
        String s=in.nextLine();
        ww.write(s);
        ww.newLine();
        Scanner sc=new Scanner(s);
        while(sc.hasNext())
            stage[++m]=sc.nextInt();
        while(in.hasNext())
        {
            s=in.nextLine();
            sc=new Scanner(s);
```

```
    name[++n]=sc.next();
    str[n]=s;
    for(int i=1;i<=m;i++)
        bb[n][i]=sc.nextDouble();
    normal(n);
}
for(int i=1;i<=m;i++)
    cnt[stage[i]]++;
for(int i=1;i<=m;i++)
for(int j=1;j<=m;j++)
    for(int k=1;k<=n;k++)
        dis[i][j]+=(aa[k][i]-aa[k][j])*(aa[k][i]-aa[k][j]);
for(int i=1;i<=m;i++)
{
    for(int j=1;j<=4;j++) rr[j]=1000000.0;
    for(int j=1;j<=m;j++)
    {
        if(j==i) continue;
        if(dis[i][j]<rr[stage[j]])
        {
            rr[stage[j]]=dis[i][j];
            mm[i][stage[j]]=j;
        }
    }
}
for(int i=1;i<=4;i++)
    rr[i]=(double)cnt[i]/m;
for(int k=1;k<=n;k++)
{
    hh=0.0;
    for(int i=1;i<=m;i++)
    {
        ll=aa[k][i]-aa[k][mm[i][stage[i]]];
        hh-=ll*ll;
        for(int j=1;j<=4;j++)
        {
            if(j==stage[i]) continue;
            ll=aa[k][i]-aa[k][mm[i][j]];
            hh+=rr[j]*ll;
        }
    }
    pp[k]=new Pair(hh,k);
}
Arrays.sort(pp,1,n+1);
```

```
        for(int i=1;i<=n;i++)
        {
            if(pp[i].x<eps) break;
            ww.write(str[pp[i].y]);
            ww.newLine();
        }
        ww.close();
    }
}
```