数据挖掘流程与类型

1. 前言

在数据挖掘领域，CRISP-DM[[1]](#footnote-1)（Cross-Industry Standard Process for Data Mining）和SEMMA[[2]](#footnote-2)（Sample, Explore, Modify, Model, and Assess）是两种常用的数据挖掘流程。KDD（Knowledge discovery in databases)）流程是表1描述了KDD[[3]](#footnote-3)流程[[4]](#footnote-4)与CRISP-DM及SEMMA对应关系。本文基于CRISP-DM来详解数据挖掘标准流程，SEMMA也会提及，两种流程内在逻辑是相近的[[5]](#footnote-5)。

表1 KDD与CRISP-DM及SEMMA对应关系概述

|  |  |  |
| --- | --- | --- |
| KDD | SEMMA | CRISP-DM |
| Pre KDD | ------ | Business understanding |
| Selection | Sample | Data understanding |
| Pre processing | Explore |
| Transformation | Modify | Data preparation |
| Data mining | Model | Modeling |
| Interpretation/Evaluation | Assessment | Evaluation |
| Post KDD | ------ | Deployment |

CRISP-DM即 “跨行业数据挖掘标准流程”， 是一种业界认可的用于指导数据挖掘工作的方法。

1996年，当时数据挖掘市场是年轻而不成熟的，但是这个市场显示了爆炸式的增长。三个在这方面经验丰富的公司**DaimlerChrysler、SPSS、NCR**发起建立一个社团，目的建立数据挖掘方法和过程的标准。在获得了EC（European Commission）的资助后，他们开始实现他们的目标。为了征集业界广泛的意见共享知识，他们创建了CRISP-DM Special Interest Group（简称为SIG）。

1999年，SIG组织开发并提炼出CRISP-DM，同时在Mercedes-Benz和OHRA进行了大规模数据挖掘项目的实际试用。2000年，CRISP-DM 1.0公开发表。这项标准的目标是支持来自不同行业的人在数据挖掘活动中使用相同的术语、方法、工具。CRISP-DM强调，DM不单是数据的组织或者呈现，也不仅是数据分析和统计建模，而是一个从理解业务需求、寻求解决方案到接受实践检验的完整过程。。CRISP-DM的成功在于它是建立在数据挖掘工程实践和实际经验基础上的合理的标准。

1. CRISP-DM 概述

CRISP-DM的主要特征是，它是一个将业务目标与分析目标结合的开放模型。数据挖掘项目生命周期包含六个阶段，参见图1。这些阶段的前后顺序并不是固定的。在不同的阶段间向前或向后移动总是必要的。该模型考虑了经常不可避免地返回数据挖掘过程的前一个阶段的因素。接下来要进行哪一步骤或一个阶段的哪项特定任务，取决于每一个阶段的结果。该项目不是一个由起点、一系列预定好的步骤、终点组成的线性过程。而是以循环性为特点。

解决方案部署以后，数据挖掘没有结束。在过程中和部署解决方时所获得的经验教训可以引发新的、常常是更加有针对性的业务问题。后面的数据挖掘过程将从前面的过程中受益。

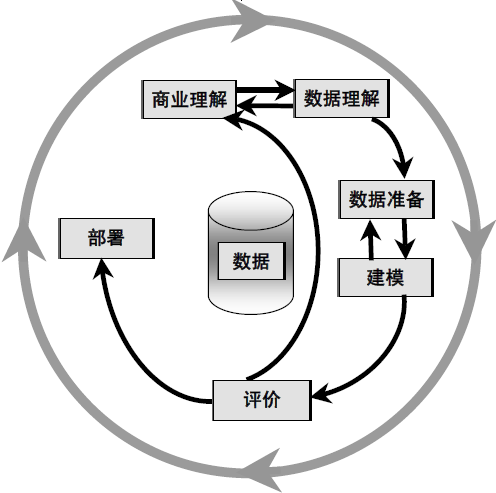


图1 CRISP-DM参考模型阶段

下面对每一个阶段进行简要介绍：

1、业务理解（Business understanding）： 这个最开始的阶段专注于从业务的角度理解项目的目标和要求，然后将这个知识转化为一个数据挖掘问题的定义和实现该目标的最初规划。

2、 数据理解（Data understanding）： 数据理解阶段以初步数据收集开始，检查数据的可访问性和解决具体的业务问题的充分性，接下来进行一些活动，目的是熟悉数据，识别数据质量的问题，从而获得关于数据的第一手信息，发现有趣的子集，形成对隐含的信息的假设。

3、数据准备（Data preparation）： 数据准备阶段覆盖了所有从初始的原始数据构造最终数据集(将要输入建模工具的数据)的活动。数据准备的任务可能多次执行，并且没有顺序规定。任务包含表格、记录和属性的选择以及为建模工具转换和清洗数据。

4、建模（Modeling）： 在这一阶段，可以选择并应用不同的建模技术，并且将参数也校准到最优的值。通常用于解决同一类型的数据挖掘问题的几个技术要经过测试，以寻找满足特定需求的最适合的技术。有些技术对数据的格式有具体的要求。因此，经常需要返回数据准备的阶段。

5、 评估（Evaluation）： 在项目的这个阶段，你已经构建了一个(或多个)从数据分析的角度来说看上去质量较高的模型。在进行模型的最终部署之前，一定要确定它正确地反映了业务的目标。关键的目标是否有一些重要的业务问题没有充分考虑到，让你必须返回到业务理解阶段。这个阶段的最后，还应该确定使用数据挖掘结果得到的决策是什么。

6、部署（Deployment）： 在创建模型过程中获得的知识可以被组织起来并以用户能够使用的方式将其呈现。数据挖掘解决方案必须像简单的静态报表一样部署给决策者，或直接写入现有的数据库(数据库评分)。模型的建立通常并不意味着项目的结束。尽管模型的目的是为了提升数据的知识力，但获得的知识需要被组织和表示成用户可用的形式。这常常与包含能支持公司决策的“现场”模型（“live” models）的使用有关，例如，Web 页面的即时个性化服务或者销售数据库的重复积分等。然而，与具体需求有关，部署阶段可认为是与生成一份报告一样简单，或者认为是与实施一个覆盖整个企业可重复的数据挖掘过程一样复杂。

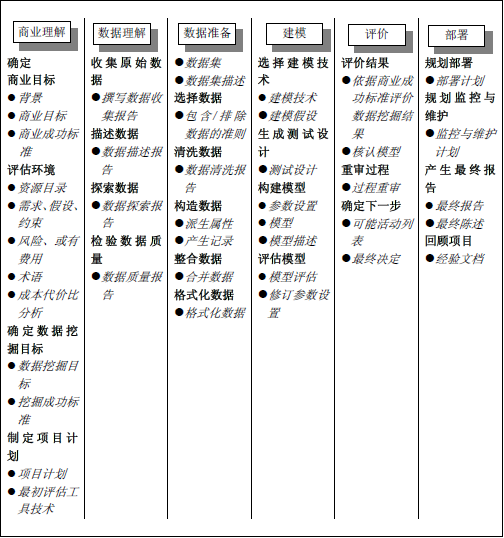


图2 CRISP-DM 参考模型的一般任务（粗体）和其输出（斜体）

图2概要描述了各阶段的一般任务（粗体）和其输出（斜体）。接下去章节，我们详细描述每一个一般任务和其输出。

1. CRISP-DM 流程

## 商业理解(Business understanding)

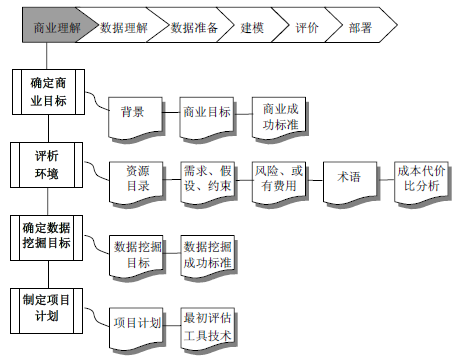


图3 商业理解

### 确定商业目标( Determine business objectives )

|  |  |
| --- | --- |
| 任务 | 确定商业目标(Determine business objectives)  数据分析师的第一个目标就是从商业角度全面理解客户真正想要达到什么目标。通常，客户有很多竞争目标，不得不正确地权衡。分析师的目标是要在一开始就揭示出能影响项目结果的重要因素。忽视这个步骤的一个可能后果是花费了大量精力却只对错误问题给出了正确答案。 |
| 输出 | 背景(Background)  背景记录了项目开始时了解到的有关公司商业环境的信息。  商业目标(Business objectives)  商业目标从商业角度描述了客户的主要目标。除了主要商业目标外，客户通常还有大量想解决的相关商业问题。比如，公司主要的商业目标是通过预测实现：当出现自己的客户倾向于跑向对手公司时，能留住他们。相关的商业问题就会是：“某银行客户的主要访问渠道（如ATM、 到支行、互联网）是如何影响他们停留的？”或“更低的ATM 费用会显著降低高价值客户留下的人数吗？”  商业成功的标准(Business success criteria)  可以从商业角度的各种观点来描述项目结果是成功的或有用的。有些标准相当具体并能被客观度量，如某种层度上客户流失的减少；还有一些一般意义但具主观性的标准，如“对关系有用的认识”。后面那种标准就应该指明是谁做出的主观判断。 |

### 评析环境( Assess situation )

|  |  |
| --- | --- |
| 任务 | 评析环境(Assess situation)  本任务涉及到更详细的事实发现：全部资源、约束、假设，及其它一些在确定数据分析目标和项目计划时也应该考虑到的因素。在前一个任务中，你的目标很快就触及到了情况的关键之处。而这里，你需要把它细节化为“有血有肉”的内容。 |
| 输出 | 资源目录(Inventory of resources)  列出项目可用的资源，包括：人员（商业专家、数据专家、技术支持人员、数据挖掘职员），数据（固定抽取的、访问现场仓库或操作型数据），计算资源（硬件平台）和软件（数据挖掘工具、其他相关软件）。  需求、假设和约束(Requirements，assumptions and constraints)  列出该项目的全部需求，包括：完成项目的时间表、结果和安全方面的可理解性和质量，还有法律问题等。作为这个输出的一部分，确信你有权使用数据。  列出该项目制定的假设。这可能是有关数据的假设，它们可以在数据挖掘过程中被检验；也可能是与作为项目基础的商业有关的假设，它们是无法检验的。后面的这种假设如果是作为结果正确性验证的前期条件时，列出它们就极其重要了。  列出该项目的约束。可能是资源可用性方面的限制，也可能是技术上的约束，比如对于建模实际使用数据的规模。  风险和或有费用(Risks and contingencies)  列出可能导致项目延期或失败的风险或事件。列出相应或有费用计划，以及风险出现时，应该采取何种行动。  术语(Terminology)  一个与该项目有关的术语表。它包括两个部分：  （1）与商业有关的术语表，它是项目进行商业理解的一部分。编订这样一个术语表是一个很有用的“知识获取”和教育培训。  （2）与数据挖掘有关的术语表，对于有疑问的商业问题还需配以实例解释。  成本和收益(Costs and benefits)  对该项目进行成本收益分析，比较项目成本与项目成功后为商业的带来的可能收益。这个比较应尽可能详细，比如,可以采用贸易中的货币指标。 |

### 确定数据挖掘目标( Determine data mining goals )

|  |  |
| --- | --- |
| 任务 | 确定数据挖掘目标(Determine data mining goals)  商业目标是以商业术语描述的，而数据挖掘目标是以技术术语描述的项目目标。例如，商业目标可能是“增加现有顾客的总销售额”。而一个数据挖掘目标则也许是“给定客户过去三年的购买信息、人口统计学信息（年龄、收入、城市等）和项目明细价格，预测客户会买多少装饰品”。 |
| 输出 | 数据挖掘目标(Datamining goals)  描述该项目的预计输出，该输出使得商业目标得以实现。  数据挖掘成功标准(Datamining success criteria)  以技术术语定义好项目结果成功的标准，比如，某种预测准确度水平或在给定“提升度”下购买兴趣的倾向。与商业成功标准一样，也许必需以主观方式来描述这些标准，此时，应该标识是哪个或哪些人作出了这个主观判断。 |

### 制定项目计划( Produce project plan )

|  |  |
| --- | --- |
| 任务 | 制定项目计划(Produce project plan)  描述为达到数据挖掘目标进而实现商业目标的确定计划。该计划应该详细列出项目后续期间需要完成的一系列步骤包括最初对工具和技术的选择。 |
| 输出 | 项目计划(Project plan)  列出项目需要经历的各个阶段，包括持续时间、需要的资源、输入、输出和关联性。这里要尽可能对挖掘过程中会大量重复的步骤交待清楚，比如建模和评估阶段的重复。  作为项目计划的一部分，分析时间进度和风险之间的关联也是很重要的。在项目计划中应该明显地标记这些分析的结果，应对风险的行动和建议。  注意：项目计划包括每个阶段的详细计划。例如，在评估阶段，某处需要使用哪种评估策略。  某种意义上讲，项目计划是一个动态文档，在每个阶段结束时，需要对进展和成果情况进行重审，由此建议对项目计划做相应地更新。同时，指出重审点也是项目计划的一部分。  工具和技术的初步评估(Initial assessment of tools and techniques)  第一阶段的结束，项目要完成一个对工具和技术的初步评估。比如，这里，你要选择一种能为过程各阶段提供多种方法的数据挖掘工具。因为工具和技术的选择可能影响整个项目，所以对其早点做评估就较为重要了。 |

## 数据理解(Data understanding)

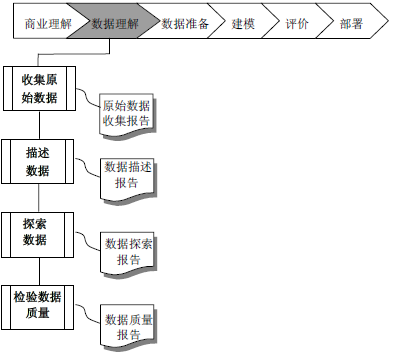


图4 数据理解

### 收集原始数据( Collect initial data )

|  |  |
| --- | --- |
| 任务 | 收集原始数据(Collect initial data)  在项目范围内，获得要列入项目资源的数据（或数据的访问方式）。初步收集包括了为理解数据而必要的数据加载(data loading)，如果你使用了一个特殊的工具来辅助数据理解，更好的做法是把数据导入到该工具中。这个做法可能就走到了原始数据准备的步骤。  注意：如果面临多个数据源，那么，不管是现在还是以后的数据准备阶段，如何对它们整合都将会是一个新的难题。 |
| 输出 | 原始数据收集报告(Initial data collection report)  列出获得的数据集(或多个数据集)，包括它们在项目中的位置，获得的方法及遇到的问题。记录遇到的问题和解决方案有助于迁移到将来项目或者推进类似项目。 |

### 描述数据( Describe data )

|  |  |
| --- | --- |
| 任务 | 描述数据(Describe data)  审查数据的“总体”或“外在”性质并给出结果的报告。 |
| 输出 | 数据描述报告(Data description report)  描述已获得的数据，包括数据格式、数据质量。比如，记录总数、各个表的字段数，字段的标识及其它被发现的外在数据特征。收集到的数据满足有关要求吗？ |

### 探索数据( Explore data )

|  |  |
| --- | --- |
| 任务 | 探索数据(Explore data)  这个任务采用查询、可视化、报告等方式解决数据挖掘中遇到的问题：关键属性比如预测任务的目标属性；一对或少量属性间的关系；简单汇总的结果；显著子聚类的特征；简单统计分析。这些分析可能直接面向数据挖掘目标，也可能有助于撰写或精炼数据描述与质量报告，以及反馈到数据转换和其他数据准备工作中做进一步的分析。 |
| 输出 | 数据探索报告(Data exploration report)  描述该任务的结果，包括初步的发现或假设以及它们对于项目后续阶段的影响。合适的话，可以把揭示数据特征的一些图表写入报告，这些图表也能导出一些令人感兴趣的数据子集以便做更进一步的检查。 |

### 检验数据质量( Verify data quality )

|  |  |
| --- | --- |
| 任务 | 检验数据质量(Verify data quality)  检查数据的质量，回答诸如下列问题：数据是完整的吗（它们覆盖全部需要考虑的情况吗）？它是正确的或者是错误的？错误有多常见？数据中有缺失值吗？如果这样问题被提出来了，那么它们出现在哪里以及出现的有多频繁呢？ |
| 输出 | 数据质量报告(Data quality report)  列出数据质量检验的结果；若存在质量问题，列出可能的解决办法。质量问题的解决办法通常很大程度地依赖于数据和商业知识。 |

## 数据准备(Data preparation)

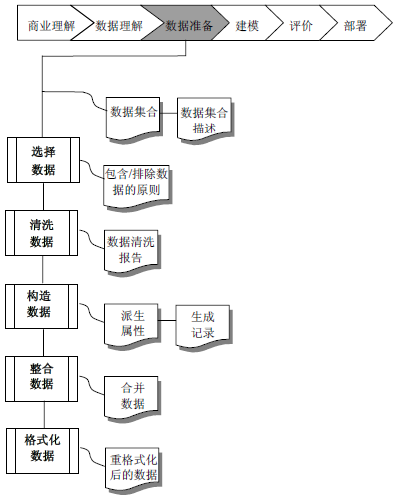


图5 数据准备

|  |  |
| --- | --- |
| 输出 | 数据集(Dataset)  数据准备阶段产生的数据集(或者一些数据集)，将用于建模或项目的主要分析工作。  数据集描述(Dataset description)  描述那些将用于建模或项目主要分析工作的数据集(或者一些数据集)。 |

### 选择数据( Select data )

|  |  |
| --- | --- |
| 任务 | 选择数据(Select data)  确定用于分析的数据。确定的标准包括：与数据挖掘目标的相关性、质量和技术限制比如数据容量或数据类型的限制。注意：数据选择包括了表中属性（列）的选择和记录（行）的选择。 |
| 输出 | 包含/排除数据的原则(Rationale for inclusion/exclusion)  列出被包含进来的和被排除出去的数据，并给出理由。 |

### 清洗数据( Clean data )

|  |  |
| --- | --- |
| 任务 | 清洗数据(Clean data)  借助选用的分析技术提升数据质量到既定层次。这涉及到数据清洗子集的选择、恰当缺省值的插入、更高级的技术如通过建模估计缺失值。 |
| 输出 | 数据清洗报告(Data cleaning report)  把在数据理解阶段由检验数据质量任务所报告的数据质量问题的解决策略和行动描述出来。为清洗目的而进行的数据转换及其对分析结果的潜在影响也在报告考虑之列。 |

### 构造数据( Construct data )

|  |  |
| --- | --- |
| 任务 | 构造数据(Construct data)  该任务包括构造性的数据准备操作，如派生属性、全新记录的生成、或现有属性的值转换。 |
| 输出 | 派生属性(Derived attributes)  派生属性是在同一记录中的一个或多个既有属性基础上构造出来的新属性。例：area=length\*width。  生成记录(Generated records)  描述全新记录的生成与创建。例：生成过去几年没有购买商品的顾客记录。原始数据中是无论如何都不存在这种记录的，但从建模目的来看，可以明白地表示存在零购买量这样一类顾客的事实。 |

### 整合数据( Integrate data )

|  |  |
| --- | --- |
| 任务 | 整合数据(Integrate data)  提供些方法，藉从多个表或记录组合成的信息可以构造出新的记录或值。 |
| 输出 | 合并数据(Merged data)  表合并是指把表示相同对象的两个或多个表合在一块。例：一个零售连锁店有一个表描述有关各个分店的一般特征（如面积、所处商业区的类型）的信息，另一个表记录有关销售的概要数据（如利率、同上一年的销售百分比变化），还有一个关于周边地段的人口统计学信息的表，这些表都有一条与每个分店相关的记录。通过把源表的字段组合在一起，这些表就可以合并成一个新表，仅用一条记录表示一个分店。  合并数据也会涉及到聚合。聚合是指通过汇总从多条记录和/或多个表的信息来计算新值的操作。比如，把一个每条记录对应每笔购买的顾客购买信息的表转换成一个新表，其中每条记录对应每个顾客，字段则是购买次数、平均购买额、使用信用卡订购的百分比、购买促销商品的比例等。 |

### 格式化数据( Format data )

|  |  |
| --- | --- |
| 任务 | 格式化数据(Format data)  格式化转换主要是指对数据进行的不改变数据含义的句法修改，这可能需要建模工具才能实现。 |
| 输出 | 重格式化后的数据(Reformatted data)  一些工具对属性顺序有特别的要求，比如第一个字段是每条记录的唯一标识，或者最后一个字段是模型需要预测的结果字段。  改变数据集中记录的顺序也许是很重要的。建模工具可能要求记录按照结果字段值排序。一般情况是，数据集记录最初是以某种顺序方式排列的，但建模算法却需要把它们以相当随机的方式排序。例如，当使用神经网络时，一般就最好使用随机排列的记录。有些工具能自动地完成这种处理，而不需用户干预。  此外，有些纯粹的对数据进行句法改变以满足特殊建模工具的要求。例如：逗号分割数据文件中把包含在文本字段内的逗号移除，把全部的值裁减到32个字符以内。 |

## 建模(Modeling)

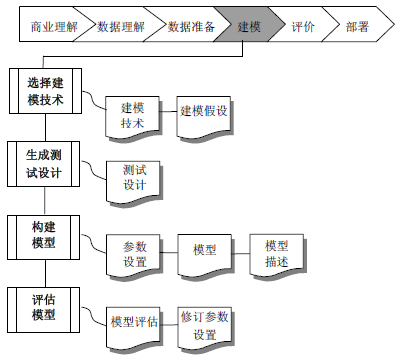


图6 建模

### 选择建模技术( Select modeling technique )

|  |  |
| --- | --- |
| 任务 | 选择建模技术(Select modeling technique)  建模的第一步，是选择将要使用的实际建模技术。尽管可能在商业理解时，你已选择过一个工具，而这里的任务却是指具体的建模技术，如由C4.5构造决策树或使用反向传播构造神经网络。若有多种技术可用，就按每种技术分别执行本任务。 |
| 输出 | 建模技术(Modeling technique)  文档化将要使用的实际建模技术。  建模假设(Modeling assumptions)  很多建模技术需要对数据做些特殊的假设，例如，全部属性具有相同的统计分布，不允许缺失值，类别属性是符号型等。对记录也要做相应假设。 |

### 生成测试设计( Generate Test design )

|  |  |
| --- | --- |
| 任务 | 生成测试设计(Generate Test design)  在实际构建模型之前，我们需要制定一个测试模型质量和有效性的程序或机制。例如，在有监督数据挖掘任务如分类中，常使用错误率作为衡量数据挖掘模型的质量指标。因此，我们一般把数据集分成训练集和测试集，在训练集上建立模型，在分开的测试集上评估模型质量。 |
| 输出 | 测试设计(Test design)  描述训练、测试和评估模型的确定计划。计划的主要部分是确定如何分割可用数据集为训练集、测试集和验证集。 |

### 建立模型( Build model )

|  |  |
| --- | --- |
| 任务 | 建立模型(Build model)  在准备好的数据集上运行建模工具，以创建一个或多个模型。 |
| 输出 | 参数设置(Parameter settings)  许多模型工具，通常都有大量需要调整的参数。列出这些参数和它们的设置值，及选择参数设置的基本原则。  模型(Models)  一些由建模工具产生的实际模型，而不是报告。  模型描述(Model description)  描述最终结果模型。生成解释模型的报告，并记下理解其含义会遇到的困难。 |

### 评估模型( Assess model )

|  |  |
| --- | --- |
| 任务 | 评估模型(Assess model)  数据挖掘工程师根据他的领域知识、数据挖掘成功标准和既定的测  试设计来解释模型。这个任务会影响到接下去的评估阶段。鉴于数据挖掘工程师判断模型应用和发现技术的成功过于技术化，那么他就应该与商业分析师和后期才出现的领域专家接触，以商业环境中的方式来讨论得到的数据挖掘结果。而且，本任务只考虑模型，然而评估阶段同时还考虑项目进程中产生的其它所有结果。  数据挖掘工程师对模型进行评定，根据评估标准来评价模型。他也尽可能地考虑商业目标和商业成功标准。在大多数数据挖掘项目中，数据挖掘工程师多次应用一个技术或者采用不同的可选技术产生数据挖掘结果。这样，在这个任务中，他也要参照评估标准来比较全部结果。 |
| 输出 | 模型评估(Model assessment)  概述本任务的结果，列出全部建成模型的质量特性（如，使用准确度术语）以及模型之间的质量等级比较次序。  修订参数设置(Revised parameter settings)  根据模型评估，修订参数设置并调整其值以完成下轮建立模型的任务。通常需要反复地进行模型建立和评估，直到你确信已找到最好模型为止。在文档中记录下所有这些修订和评估。 |

## 评价(Evaluation)

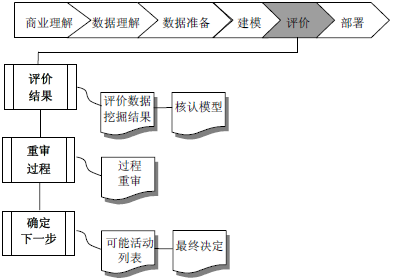


图7 评价

### 评价结果( Evaluate results )

|  |  |
| --- | --- |
| 任务 | 评价结果(Evaluate results)  前面的评估步骤处理的是如模型准确度和一般性等因素。这里的步 骤评价的是模型适合商业目标的程度并找到一些商业理由来说明某个模 型的不足。在时间和预算许可的情况下，另一个非必需的评估是在实际 应用中同步运行测试应用来测试模型。  而且评估过程中也评价其它数据挖掘的结果。从模型得到的数据挖 掘结果必须与最初商业目标相关，而其它一些发现内容却不必与它有关， 但是却能为将来揭示一些额外的难处、信息、暗示。 |
| 输出 | 根据商业成功标准评价数据挖掘结果(Assessment of data mining  results with respect to business success criteria)  使用商业成功标准术语概述结果的评价，包括一个项目是否已满足既定商业目标的最终声明。  核认模型(Approved models)  有关商业成功标准的模型评价之后，满足给定标准的建成模型就成为被核准认可的模型了。 |

### 重审过程( Review process )

|  |  |
| --- | --- |
| 任务 | 重审过程(Review process)  到这里，得到的结果模型似乎有望令人满意和符合商业需要了。现在对数据挖掘项目合同做一个全面的重审是很适时的，以确定是否有任何重要因素或任务被无意识地忽略了。重审也涉及一些质量确认问题，如我们正确地建立了模型吗？我们仅使用了那些许可使用的且将来分析也可用的属性吗？ |
| 输出 | 过程的重审(Review of process)  概述过程重审并特别注明被忽略的活动和/或应该重复的活动。 |

### 确定下一步 ( Determine next steps )

|  |  |
| --- | --- |
| 任务 | 确定下一步(Determine next steps)  根据评价的结果和过程重审，要确定项目这个阶段该如何推进，需要决定是结束并适时进入部署阶段还是继续重复前面步骤或者创建新的数据挖掘项目。本任务也包括了影响决策的遗留资源和预算的分析。 |
| 输出 | 可能活动列表(List of possible actions)  列出潜在的进一步活动，并给出支持和反对每个的理由。  最终决定(Decision)  描述有关如何合理推进的最终决定。 |

## 部署(Deployment)

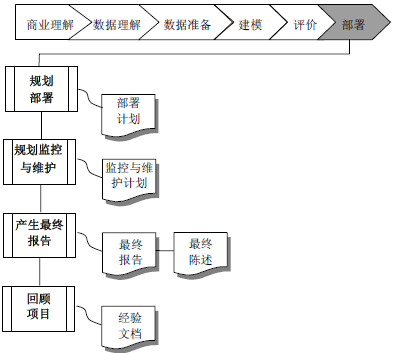


图8 部署

### 规划部署( Plan deployment )

|  |  |
| --- | --- |
| 任务 | 规划部署(Plan deployment)  为把数据挖掘结果部署到商业环境，本任务利用评估结果并给出部署的策略。如果某个一般性程序已经被认为可以创建相关模型，则为了后面的部署，要在文档中记录下该程序。 |
| 输出 | 部署计划( Deployment plan )  概述部署策略，包括必要的步骤及如何执行这些步骤。 |

### 规划监控和维护( Plan monitoring and maintenance )

|  |  |
| --- | --- |
| 任务 | 规划监控和维护(Plan monitoring and maintenance)  数据挖掘结果成为日常运作的商业及其环境的一部分时，监控和维护就成为重要问题。对维护策略的一个细致准备有助于避免数据挖掘结果长期被不正确地应用。为监控数据挖掘结果的部署，项目需要一个详细计划来监控过程。该计划需要考虑部署的具体类型。 |
| 输出 | 监控和维护计划( Monitoring and maintenance plan )  概述监控和维护策略包括必要的步骤和如何执行这些步骤。 |

### 生成最终报告( Produce final report )

|  |  |
| --- | --- |
| 任务 | 生成最终报告(Produce final report)  项目的结束，项目领导和其小组撰写一份最终报告。依赖于部署计划，这份报告可能仅对项目和其经历进行了概述（因某一活动还在进行中，它们没有被记录文档），也可能是一份最终的对数据挖掘结果全面展示的报告。 |
| 输出 | 最终报告( Final report )  这是一份关于数据挖掘项目合同的最终书面报告。它包括前面那些可交付的成果和描述以及结果组织。  最终陈述(Final presentation)  这里通常召开一个关于项目总结的会议，把结果口述给客户。 |

### 回顾项目( Review project )

|  |  |
| --- | --- |
| 任务 | 回顾项目(Review project)  评论什么是对的，什么是错的，什么做了，什么需要改进。 |
| 输出 | 经验文档( Experience documentation )  描述项目期间获得的重要经验。例如，缺陷、易误解的方法、在类似情况中选择最佳数据挖掘技术的提示等都可以写入文档。在更理想的项目中，经验文档也包括项目成员个人撰写的有关项目阶段和各自任务的报告。 |

1. 数据挖掘类型

数据挖掘模型大致可分成两种类型：

* **探索型模型：**在机器学习领域称为无监督学习，数据中只含有分析变量，没有自变量与因变量的角色之分。此类模型多用于探索高维数据的结构，返回内在的关联与规律。常用场景包括用户标签（用户细分），协同过滤产品推荐，产品关联分析等。
* **预测型模型：**在机器学习领域称为有监督学习，数据中含有自变量与因变量两种角色截然不同的部分。通过对历史数据建模，寻求自变量与因变量之间的函数关系。当函数关系被确定下来后，可用于结合新的自变量进行模型打分，预测因变量的取值（范围）。常用场景包括产品营销响应模型，客户流失预警模型等。

## 探索型模型

### 细分(Segmentation)

数据挖掘问题类型细分的目标是把数据分割为有意义且值得关注的子集或类。一个子集内的所有成员表现相同特征。例如，在购物篮分析中，可以依据货篮所装商品定义货篮划分。

细分可以手工或（半）自动地完成。分析师根据基于先验知识或数据描述和总结的结果对商业问题提出某些分组的设想。然而，也有自动聚类技术能发现待细分数据中还未被引起注意的和隐含结构。

细分可能作为一个单独的数据挖掘问题类型，那么细分的发现就成为数据挖掘的主要目的。例如，具有比平均年龄和收入更优先的邮政编码地址可以选用于家庭看护保险的邮递广告。

然而，通常，细分一般是解决其它问题类型的一个必需步骤。因此，其目的就成为保持可管理数据的规模或找到容易分析的同质数据子集。一般大数据集中各种影响相互交迭使得有趣的模式含糊不清。这样合适的细分就使该任务变得容易。例如，在百万条货篮记录中分析关联性就很困难。在值得关注的购物篮聚类中，例如高价值货篮、有便宜商品的货篮或特殊时间内的货篮中识别关联就容易得多（一般也更有意义）。

适合的技术：

* 聚类技术。
* 神经网络。
* 可视化。

例子：

一个汽车公司常规地收集了一些客户的社会经济特征如收入、年龄、性别、职业等的信息。使用聚类分析，公司把客户分成容易理解的子组并分析每组的结构特征。对每组采用不同的具体营销策略。

### 关联分析(Association analysis)

关联分析包含建立描述数据项目或事件之间有意义的关联的模型。在给定其它数据项目信息情况下，依赖性可用于预测数据项目的值。尽管依赖性可用于预测建模，但它们大多用于理解。依赖性可能是确定的也可能是概率意义上的。

关联是一种特殊的依赖性，最近已经广为人知了。关联描述了数据项目之间的密切关系（例如，频繁一块出现的数据项目或事件）。一个应用关联的典型场合是购物篮分析。如“在30％的购买中，啤酒和花生在一起被买”的规则就是典型关联例子。

探查关联的算法很快速并能产生很多关联。选择最令人兴趣的一个也是难题。

关联分析与预测和分类联系密切，其中关联分析隐式用于预测模型的表达。与概念描述也有联系，通常突出依赖性。

实际应用中，关联分析常与细分同时进行。对大数据集，依赖性很少是明显的，因为很多影响因素相互交迭在一起。这种情况下，在数据同质细分基础上执行关联分析就比较可取。

序列模式是一种考虑了事件顺序的特殊依赖性。在购物篮情况中，关联描述了某个特定时间的依赖。而序列模式描述的是某个顾客或顾客群体随时间变化的购物模式。

适合的技术：

* 相关分析。
* 回归分析。
* 关联规则。
* 贝叶斯网络。
* 归纳逻辑规划。
* 可视化技术。

例子1：

使用回归分析，一位商业分析师能发现在某产品的总销售额与其价格和广告总费用之间有显著关联。这样，分析师一旦发现知识，就能通过改变价格和/或相应广告支出来达到想要的销售水平。

例子2：

对有关汽车零配件的数据使用关联规则算法，汽车公司能发现：在全部记录情况中，有95%表现为如果订购了一个无线电装置，那么一个自动变速箱也会被订购。基于这个依赖性，汽车公司就可确定提供这些配件的组合来减少成本。

## 预测型模型

### 分类(Classification)

分类假设有一组对象集合，由某些属性或特征标识，每个对象属于不同的类。类标签是离散（符号）值并且对某个对象而言是确定的。目标是建立分类模型（有时称为分类器）来为未知和未加标签的对象分配类标签。分类模型也经常用于预测模型。类标签可以预先给定，例如由用户定义或从其它细分中获得。

分类是最重要的数据挖掘问题类型之一，常广泛出现在各种应用领域。很多数据挖掘问题都可以转换为分类问题。例如，信用卡积分以评估新客户的信用风险，这可转换成一个分类问题：信用好的客户和信用差的用户两个类。一个分类模型可以从现有顾客数据及其信用行为中建立。这个分类模型也可用于分类新的潜在客户是信用好还是差，以决定是接受还是拒绝他。

分类问题也几乎与所有其它问题类型相联系。因为分箱（binning）技术可以用于转换连续值域为离散间隔区间，故通过分箱连续类标签，预测问题就可转换成分类问题。这些离散间隔区间就被作为类标签而不是使用精确的数值、值了，因此变成了一个分类问题。有些分类技术也产生可理解的类或概念描述。因为分类模型一般揭示和阐明属性间的关联，所以分类也与关联分析有关。

只要建立了好的分类模型，细分就能提供类标签或限制数据集。在分类模型建立之前，偏差分析是有用的。偏差和独立点会隐藏好的分类模型的模式。另一方面，分类模型也用于识别偏差和数据中的其它问题。

适合的技术：

* 差别分析。
* 规则归纳方法。
* 逻辑回归。
* 决策树学习。
* 神经网络。
* K近邻。
* 基于实例的推理。
* 遗传算法。

例子：

银行一般有信用卡持有人的支付行为信息。把这些金融信息与客户的其它信息如性别、年龄、收入等组合在一起，就有可能设计一个系统来分类新客户为好或差，（例如，可接受客户的信用风险为低或高）。

### 预测(Prediction)

另一种广泛应用的重要问题类型是预测。预测很类似于分类。唯一的不同就是在预测中目标属性（类）不是定性的离散属性，而是连续的属性。预测目标是确定新对象的目标属性数值。这个问题类型有时称为回归。

适合的技术：

* 回归分析。
* 回归树。
* 神经网络。
* K近邻。
* 博克斯－詹金斯（Box-Jenkins）方法。
* 遗传算法。

例子：

一个国际公司的年收入与其它属性如广告、兑换率、通货膨胀率等相关。有了这些值（或值的可靠估计），公司就能预测下一年的期望收入。

通常，数据挖掘项目包含一组不同的问题类型，它们合在一起共同解决商业问题。

# 数据挖掘技术

数据挖掘常用技术包括聚类分析算法、关联规则算法、神经网络算法、决策树算法、逻辑回归算法、线性回归算法、Naive Bayes 算法及时间序列算法等。

## 聚类分析算法

定义：聚类算法的目的是将对象的群体分成有若干个类，使得同类对象之间距离较近（相似性较强），而不同类对象距离较远（相似性较弱或没有）。同类对象的距离越近，不同类对象的距离越远，表明聚类效果越好。

按聚类的过程，算法可分为基于原型的、基于层次的、基于密度的；按类的相互关系，算法可分为互斥的、重叠的与模糊的；按对象到组的归属关系，算法可分为完全的与部分的。

基于原型的聚类算法考虑每个对象在空间中的分布，每个类被一个原型（如重心、神经元、标志向量）所标识，对象经过多次迭代被分配到距离最近的原型所标识的那一类中，原型本身也随着迭代过程不断更新。基于层次的聚类可进一步细分为自下而上聚合或者自上而下拆分，前者不断聚合距离最近的个体或类；后者不断拆分距离最远的个体或类。无论采用哪种，类之间都表现出层次包含关系，因而得名。基于密度的算法考虑对象在空间的聚集密度，识别高密度区域和稀疏区域。第4节会对以上三类聚类算法做更详尽论述。

互斥的聚类约定每一对象只能归属于一类，而重叠的聚类则允许对象同属两个或多个类。模糊聚类也允许一个对象归属一个或多个类：每个对象以权重属于任何一类，亦可视为该对象属于每一类的概率。算法要求同一个对象的权重之和等于1。模糊聚类可以转换为互斥或者是重叠的结果。对于互斥的，把对象分配到具有最大权重的类；对于重叠的，把对象分配到权重大于某一预设阈值的所有类。

完全聚类要求每个对象都归属到某一类，而部分聚类无此要求。实际应用中数据往往被噪声影响，它们理应不归属到任何类。使用者有时仅关注高度聚集的一个或几个子群体，而并不在乎是否要把每个对象都一一划分到类中，导致部分聚类的产生。在评估部分聚类的效果时，亦不以全局的聚合程度作为评判标准。

篇幅关系，本文只讨论互斥的完全聚类。模糊聚类可以参考Höppner等人的论著[[6]](#footnote-6)。

### 常用距离度量

任何一种聚类算法都基于对象之间的距离度量。两个对象与的距离的度量方法多种多样，但基本满足以下特征：

非负性：

同一性：当且仅当

对称性：

三角不等性：

聚类文献中还经常使用相似性一词。相似性与距离含义相近，方向相反。大量单调函数都可以实现相似性与距离二者相互转换。

下文列举一些常见的距离度量方法。

* 闵可夫斯基距离

定义向量与之间的闵可夫斯基距离为：

上式中的是一自然数。当取不同的值时对应3种最常用的特例：

绝对值距离（Manhattan Distance, ）：

欧几里德距离（Euclidean Distance, ）：

欧几里得距离因其简单的计算、直观的效果，成为聚类算法中使用率最高的距离度量方法。

切比雪夫距离（Chebyshev distance, ）：

* 余弦距离

两个向量与的余弦距离定义为：

其中

余弦距离常被用于文本聚类。文本聚类中关注两篇文档共用的词汇高于两篇文档均不使用的词汇，余弦距离恰有此特性。

* 相似系数

两个向量与的Pearson相似系数定义为：

其中

Pearson相似系数介于，若满足，相似系数取最大值或最小值；若和相互垂直取0。Pearson相似系数忽略数据自身的取值范围而关注相互之间的线性相关程度，在聚类算法中亦有广泛应用。

Pearson相似系数易受离群点影响，基于秩的Spearman相似系数在这方面要稳定一些。其定义为：

其中

即两个维度的秩之差。

Spearman相似系数也介于，当且仅当两个对象所有维度单调同向或者单调异向时取最大值和最小值。

* 其它距离度量

以上所有距离度量方式都没有考虑向量各个维度之间的关联，为改善这一点产生了马氏距离（Mahalanobis Distance）[2]：

其中是数据各维度之间的协方差矩阵。

此外还有度量离散变量的简单匹配系数（Simple Matching Coefficient, SMC）、Jaccard系数和Tanimoto系数等，因篇幅关系不一而足。

在实际计算距离的过程中，有时会给向量的不同维度赋予不同的权重，此时向量与的距离为：

即每一维度距离的加权平均。其中为任意距离度量函数。

距离度量方式没有优劣之分，只有适应与不适应应用场景的区别。如果关注的是向量的绝对位置，以欧几里得距离为代表的方法更适合；如果关注的是向量之间的相互关系，余弦距离或相似系数显然比欧几里得距离更恰当。

### 常用聚类算法

如上文所述，聚类算法可分为基于原型的、基于层次的、基于密度的。本节介绍三类中最常用的一些算法。

#### 原型聚类

原型指可以描述类特征的元素，如重心、标志向量或神经网络中的神经元。基于原型的聚类算法中，K-Means和SOM是最广泛应用的两个。

1）K-MEANS

K-Means算法最先由MacQueen等于1967年提出[3]，算法试图将个对象划分到个类中，使得每个对象属于重心距离它最近的那一类。聚类的目标函数为：

K-Means求解是N-P问题，但利用启发式算法经常可以快速收敛并达到局部最优。常用的Lloyd实现流程为[4]：

|  |  |
| --- | --- |
| 初始化 | 给定个初始重心。 |
| 分配 | 将每个对象分配到距离最近的重心所代表的那一类，多数实现默认使用欧几里得距离： |
| 更新 | 重新计算每一类的重心： |
| 迭代 | 重复“分配——更新”过程直至对象到类的归属关系不再改变，或者达到预设阈值（如最大迭代次数）。 |

给定初始重心的方法众多，常用的包括（1）随机给定，（2）尽量使初始重心尽量分散，（3）使用其它算法的输出。由于（1）的随机性往往导致聚类结果不稳定，实际应用中（2）和（3）较受欢迎。

Lloyd算法的计算复杂度为，其中是样本量，是向量的维度，是类别数，是迭代次数。大量实验证明如果数据中确实存在类结构，算法多在10余次迭代即可达到收敛，因此Lloyd算法可以认为是线性的。

K-Means算法有其局限性，包括：（1）类数需要预先设定，（2）Lloyd实现产生的局部最优结果有时与主观认识相悖，（3）对类内对象的分布隐含球形假设，（4）对离群点敏感。尽管如此，由于Lloyd实现的高效性，K-Means仍不失为最常使用的聚类算法之一。

2）SOM

SOM（Self Organization Map）由Kohonen等于1982年提出[5][6]，是一种**基于神经网络结构的数据降维与聚类的方法**。对象被分配到距离最近的神经元所代表的类中。区别于K-Means，SOM引入神经元的拓扑结构，这一结构是预设的。在对象不断分配的过程中，距离最近神经元以及它的邻居都被更新。过程如下：

|  |  |  |
| --- | --- | --- |
| 初始化 | | 设定神经元的拓扑结构，并随机设定每个神经元的值： |
| 分配 | | 有重复的随机抽取一个对象，定位与其距离最近的神经元。即： |
| 更新 | | 对拓扑结构中的每个神经元，更新其代表的值：  其中有多种选择形式，如：  高斯核函数  气泡核函数  其中与都是随时间衰减的单调函数，是神经元在拓扑结构中的位置，预设阈值控制邻居的范围。 |
| 迭代 | 重复分配——更新流程，直至满足以下条件：   * 神经元的值收敛或变动小于预设阈值 * 达到预设最大迭代次数 | |

同样是更新——迭代过程，K-Means记录每个对象到类的归属关系，而SOM只是用它来更新神经元的值而不保存。分配过程中的有重复随机抽样可能导致有些对象被多次选取，而有些对象在聚类终止前从未被选取过。

在更新步骤中，使用高斯核函数使得每个神经元都朝着输入对象的方向更新，更新幅度随该神经元和距离最近神经元的距离变远而快速衰减；使用气泡核函数将所有神经元简单二分：距离最近神经元的邻居与非邻居，每次只更新邻居的值。

SOM的收敛速度取决于核函数的设定，往往慢于K-Means算法且不保证收敛，因此预设最大迭代次数是必要的。SOM的计算复杂度为，其中是神经元的个数，是迭代次数。

#### 层次聚类

层次聚类具有悠久的历史，最早多应用于生物分类学，主要贡献者是Jardine，Sibson和Sneath，Sokal[7][8]。层次聚类可分为不断把小类聚合成大类（自底向上）或是不断把大类拆分成小类（自顶向下），因保持了聚合、拆分过程中的层次关系而得名。

1）平均链接

平均链接（Single Linkage）又名UPGMA（Unweighted Pair Group Method with Arithmetic Mean），算法过程如下：

|  |  |
| --- | --- |
| 初始化 | 每个对象自成一类，计算类与类之间的距离。 |
| 合并 | 将距离最近的两类聚合，更新距离矩阵。两个类与之间的距离定义为： |
| 迭代 | 重复上一过程直至最终聚合成1类，或在达到一定条件时（如最近距离超出预设阈值）终止聚合获得此时的类。 |

平均链接算法复杂度为，无法应用于较大的数据集是层次聚类算法被诟病的主要原因。

2）其它层次聚类

与平均链接类似的，通过改变类与类之间距离定义产生层次聚类的各种变种，包括：

* **单链接（Single Linkage）：**类之间的距离为类间对象之间距离的最小值，即
* **全链接（Complete Linkage）：**类之间的距离为类间对象之间距离的最大值，即。
* **重心链接（Centroid Linkage，又名UPGMC）：**类之间的距离为类重心之间的距离，即，其中分别是类与类的重心。
* **Ward方法[9]：**将聚合后对总体类内方差增幅最小的两类合并。

以上所有的算法都满足一共同特征，类与类合并后产生的新类，与另一类之间的距离满足Lance-Williams公式[10]：

每种算法的Lance-Williams公式系数如表 1所示：

表 1 各种层次聚类算法的Lance-Williams公式系数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **算法** |  |  |  |  |
| 平均链接 |  |  |  |  |
| 单链接 |  |  |  |  |
| 全链接 |  |  |  |  |
| 重心链接 |  |  |  |  |
| Ward方法 |  |  |  |  |

#### 密度聚类

DBSCAN（Density-based spatial clustering of applications with noise）由Ester等于1996年提出[11]，是密度聚类中最常用的算法。DBSCAN对数据库应用有良好的支持。

1）DBSCAN

DBSCAN在空间中将密度较高的点集（对象群体）聚合。准确地说，DBSCAN将空间中的点分为核心点、可到达点与离群点三类。聚类过程如下：

|  |  |
| --- | --- |
| 标记 | * 核心点：核心点是周围拥有较多邻居的点。邻居指以核心点为圆心，为半径的球体内的点；较多指超过预设阈值。这些邻居被认为是从核心点可“直接到达”的。 * 可到达点：可到达点不是核心点，但存在一条路径满足：从可直接到达。 * 离群点：非核心点与可到达点为离群点。离群点“不可到达”。 |
| 归类 | 完成标记后，每个核心点和它可到达的点集构成一个类。一类可包含一或多个核心点。 |

DBSCAN的算法复杂度介于到，取决于在半径内寻找邻居的复杂度，与实现的数据结构相关。它相对于K-Means算法具有很多优势：（1）DBSCAN不需要预先指定类别数，（2）DBSCAN可以发现任意形状的类，（3）DBSCAN能够识别离群点（噪声）。

DBSCAN并非完美，它不适用于密度变化较大的数据集，从理论上无法找到单一的与适用于不同的密度区域。

## 聚类算法评估

不管数据自身是否存在类结构，上述聚类算法总能找到一种划分对象到类中的方法。不同的算法产生迥异的结果，因此聚类的评估很有必要。评估分为无监督（内部评估准则）与有监督（外部评估准则）两种。无监督评估仅使用数据自身的信息评估聚类效果，而有监督的评估对比聚类结果与数据之外的组别标记信息的匹配程度。

### 无监督评估

约定是一个聚类评估函数，众多无监督评估准则可以描述为：

即总体的评估效果是每一类评估效果的加权平均。评估函数可以基于：

* 类内分散程度：或，其中为类的重心。
* 类间分散程度：或或，其中为所有对象的重心。

或者二者的组合。好的聚类应当使类内的分散程度尽量低，同时类间的分散程度尽量高。常用的评估准则如表 2所示：

表 2 常用无监督评估准则

|  |  |
| --- | --- |
| 评估函数 | 权重 |
|  |  |
|  | 1 |
|  |  |
|  |  |

除了形如的评估函数，还存在其它形式的评估指数。Silhouette和Dunn指数是其中使用率最高的两个[12][13]。

Silhouette计算过程中也同时涵盖了类内与类间的分散程度。对于任意一个对象，计算它到同类所有其它对象的距离，记为：

分别计算对象到其它各类内每个对象的平均距离，作为对象到各类的距离，这些距离中最小的记为。

对于对象，它的Silhouette指数定义为：

Silhouette指数的值域是，负的指数暗示对象的分配或存在问题（它到另一类的距离更近）。单个对象的Silhouette指数可以汇总平均为每个类的指数，也可以进一步汇总平均为整体聚类的指数：

Dunn指数形式为：

其中与分别是类内与类间分散程度的评估函数，可以取上述的任一形式。Dunn指数越大，越代表好的聚类效果。注意到分母的函数形式使得Dunn指数对个体敏感。设想聚类结果由个紧密的类和1个松散的类构成，该指数会受到这个松散类的严重影响而分值较低。

### 有监督评估

有监督评估指拥有数据额外的分组标签时，判断聚类结果与外部分组标签匹配程度的过程。下文使用“组”描述外部标签，使用“类”描述聚类结果以区分。常用的评估指数包括熵、Rand和Jaccard指数等[14][15]。

对于任意一类，计算组在其中的分布概率，定义：

其中是类中的对象个数，是组在类中的对象个数。该类的熵定义为：

其中是总组数。总体的熵可视为各类熵的加权平均，权重为每类的对象个数：

熵越小表明类与组之间的一致性越强，反之亦反。

以Rand和Jaccard指数为代表的另一类评估方法是统计所有的对象两两之间的分组、分类关系，并汇总成下面四个统计量：

1. ：两个对象既不在同一组也不在同一类的总数；
2. ：两个对象不在同一组，但在同一类的总数；
3. ：两个对象在同一组，但不在同一类的总数；
4. ：两个对象既在同一组也在同一类的总数。

定义Rand指数为：

定义Jaccard指数为：

Jaccard指数相对Rand指数只是没有计入的影响，适用于聚类对象数目庞大而使用者更关注匹配度的情况。

### 类数的确定

如上文所述的K-Means或各个层次聚类算法，需要预先给出类数，在现实应用中较为困难：研究者往往不知道数据的内在结构，或数据中本没有清晰的类。此时多借助聚类评估准则，在一定范围内寻找合适的值使得评估效果最好。过多的类数使得结果解释繁琐且困难，应用中常把的上限设定在范围内。

### 各种聚类算法优缺点对比

表2 各种聚类算法优缺点对比

|  |  |  |
| --- | --- | --- |
| **算法** | **优势** | **劣势** |
| K-Means | * 快速，适用于大规模数据 | * 类数需要预先设定 * 局部最优非全局最优 * 类结构球形假设 * 对离群点敏感 |
| SOM | * 数据降维与可视化，有助于探索数据结构 | * 较多的预设参数 * 迭代次数较高，不保证收敛 * 没有目标函数，不同SOM聚类的结果不易直接比较 |
| UPGMA等层次聚类 | * 聚类过程存在层次关系，有助于探索数据结构 | * 时间复杂度高 * 类数需要预先设定 |
| DBSCAN | * 不需要预先指定类别数 * 对类的形状没有要求 * 可以识别离群点 | * 邻居个数与邻居半径需要预设 * 要求数据密度差异性不大 * 不适用于高维数据 |

## 朴素贝叶斯算法

朴素贝叶斯是一种构建分类器的简单方法。该分类器模型会给问题实例分配用特征值表示的类标签，类标签取自有限集合。它不是训练这种分类器的单一算法，而是一系列基于相同原理的算法：所有朴素贝叶斯分类器都假定样本每个特征与其他特征都不相关。 举个例子，如果一种水果其具有红，圆，直径大概3英寸等特征，该水果可以被判定为是苹果。尽管这些特征相互依赖或者有些特征由其他特征决定，然而朴素贝叶斯分类器认为这些属性在判定该水果是否为苹果的概率分布上独立的。

对于某些类型的概率模型，在有监督学习的样本集中能获取得非常好的分类效果。在许多实际应用中，朴素贝叶斯模型参数估计使用最大似然估计方法；换言之，在不用贝叶斯概率或者任何贝叶斯模型的情况下，朴素贝叶斯模型也能奏效。

尽管是带着这些朴素思想和过于简单化的假设，但朴素贝叶斯分类器在很多复杂的现实情形中仍能够取得相当好的效果。尽管如此，有论文证明更新的方法（如提升树和随机森林）的性能超过了贝叶斯分类器。

朴素贝叶斯分类器的一个优势在于只需要根据少量的训练数据估计出必要的参数（变量的均值和方差）。由于变量独立假设，只需要估计各个变量，而不需要确定整个协方差矩阵。

**朴素贝叶斯的优缺点**

**优点：**

学习和预测的效率高，且易于实现；在数据较少的情况下仍然有效，可以处理多分类问题。

**缺点：**

**分类效果不一定很高，特征独立性假设会是朴素贝叶斯变得简单，但是会牺牲一定的分类准确率。**

## 决策树算法

定义：机器学习中，决策树是一个预测模型；他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。决策树仅有单一输出，若欲有复数输出，可以建立独立的决策树以处理不同输出。 数据挖掘中决策树是一种经常要用到的技术，可以用于分析数据，同样也可以用来作预测[[7]](#footnote-7)。

决策树算法有很多种，根据属性分割方法的不同决策树算法分为两类：基于信息论的方法：例如ID3、C4.5算法；基于最小Gini指标方法：SLIQ、SPRD等算法。ID3算法是国际上公认的最早有影响的决策树算法。ID3算法根据属性集的取值进行分类的决策树算法，采用自顶向下不回溯的策略搜索全部的属性空间，它建立决策树的算法简单，分类速度快；但是ID3对于大的属性集则执行效率低，准确性不够，并且学习能力低下。C4.5算法继承了ID3算法的全部优点，相对于ID3算法C4.5算法增加了增益比例的概念，可以处理连续属性以及属性值不完整的训练样本等。同时也存在许多缺点，比如：得到的结果并不是全局最优的，C4.5算法构造出来的决策树很难改善，C4.5算法对属性进行分组的效率比较低。为了解决决策树算法(例如C4.5算法)要求训练数据驻留内存从而不适合处理大规模数据的问题，IBM的研究人员提出了SLIQ算法。SLIQ算法是一种高速可伸缩的数据挖掘分类算法。SLIQ算法采用类直方图的数据结构对属性进行分片，这种类直方图要一直驻留在内存，类直方图结构的大小由训练样本数成正比关系，因此当数据集很大是算法的效率非常低而且有可能失效。为了解决SLIQ算法中类表必须驻留在内存中的缺陷，IBM的研究人员提出了sprint算法。由于该算法具有完全不受内存的限制、生成的决策树较为紧凑和准确、易于实现并行化、以及较好的伸缩性、加速性和扩容性等优点，成为了数据挖掘中优秀的算法之一。Sprint算法是一种可以扩展可以并行的归纳决策树。它吸收了SLIQ算法的预排序技术，使用了不同的数据结构，从而消除了内存的限制。SPIUNT算法包括SPIUNT串行算法和SPRINT并行算法。在实际应用中也通常把SLIQ，SPRINT通过改进使之成为分布式的决策树算法。

近几年对SPRINT的优化主要集中在以下几个方面。（１）减少计算量提高计算效率。每次求最佳分割点时sprint算法都要对所有候选属性的每一个候选分割点来计算gini值，计算量大。针对离散属性和连续属性分别提出不同的优化方案（２）使用雨林算法框架，减少通信量，提高易扩展性。（３）引入新的数据结构提高资源利用率，减少I/O和扫描磁盘的次数。

Sprint算法的优势是通过增加大量磁盘L／0读写时间和搜索磁盘数据时间来获得的。正是因为这个特点，SPRINT方法在非常大的数据集上构造决策树的计算时间会很长，以至缺乏实际意义。’处理这个问题有两种较好方式：一种是采用并行性机制，另一种是减少算法的I／O时间。最近几年随着分布式及云计算技术的逐渐成熟，有许多研究工作人员将并行sprint算法改进应用到分布式平台（比如Hadoop）之上。所以随着云计算和数据挖掘技术的研究深入，SPRINT算法的优势将逐步得到体现。

**决策树算法的优缺点：**

**优点：**

1 决策树易于理解和解释；

2 能够同时处理数据型和类别型属性；

3 决策树是一个白盒模型，给定一个观察模型，很容易推出相应的逻辑表达式；

4 在相对较短的时间内能够对大型数据作出效果良好的结果；

5 比较适合处理有缺失属性值的样本。

**缺点：**

1 对那些各类别数据量不一致的数据，在决策树种，信息增益的结果偏向那些具有更多数值的特征；

2 容易过拟合；

3 忽略了数据集中属性之间的相关性。

## 逻辑回归算法

逻辑回归又称logistic回归分析，是一种广义的线性回归分析模型，常用于数据挖掘，疾病自动诊断，经济预测等领域。例如，探讨引发疾病的危险因素，并根据危险因素预测疾病发生的概率等。以胃癌病情分析为例，选择两组人群，一组是胃癌组，一组是非胃癌组，两组人群必定具有不同的体征与生活方式等。因此因变量就为是否胃癌，值为“是”或“否”，自变量就可以包括很多了，如年龄、性别、饮食习惯、幽门螺杆菌感染等。自变量既可以是连续的，也可以是分类的。然后通过logistic回归分析，可以得到自变量的权重，从而可以大致了解到底哪些因素是胃癌的危险因素。同时根据该权值可以根据危险因素预测一个人患癌症的可能性。

logistic回归是一种广义线性回归（generalized linear model），因此与多重线性回归分析有很多相同之处。它们的模型形式基本上相同，都具有 w‘x+b，其中w和b是待求参数，其区别在于他们的因变量不同，多重线性回归直接将w‘x+b作为因变量，即y =w‘x+b，而logistic回归则通过函数L将w‘x+b对应一个隐状态p，p =L(w‘x+b),然后根据p 与1-p的大小决定因变量的值。如果L是logistic函数，就是logistic回归，如果L是多项式函数就是多项式回归。

logistic回归的因变量可以是二分类的，也可以是多分类的，但是二分类的更为常用，也更加容易解释，多类可以使用softmax方法进行处理。实际中最为常用的就是二分类的logistic回归。

Logistic回归模型的适用条件

1 因变量为二分类的分类变量或某事件的发生率，并且是数值型变量。但是需要注意，重复计数现象指标不适用于Logistic回归。

2 残差和因变量都要服从二项分布。二项分布对应的是分类变量，所以不是正态分布，进而不是用最小二乘法，而是最大似然法来解决方程估计和检验问题。

3 自变量和Logistic概率是线性关系

4 各观测对象间相互独立。

原理：如果直接将线性回归的模型扣到Logistic回归中，会造成方程二边取值区间不同和普遍的非直线关系。因为Logistic中因变量为二分类变量，某个概率作为方程的因变量估计值取值范围为0-1，但是，方程右边取值范围是无穷大或者无穷小。所以，才引入Logistic回归。

Logistic回归实质：发生概率除以没有发生概率再取对数。就是这个不太繁琐的变换改变了取值区间的矛盾和因变量自变量间的曲线关系。究其原因，是发生和未发生的概率成为了比值 ，这个比值就是一个缓冲，将取值范围扩大，再进行对数变换，整个因变量改变。不仅如此，这种变换往往使得因变量和自变量之间呈线性关系，这是根据大量实践而总结。所以，Logistic回归从根本上解决因变量要不是连续变量怎么办的问题。还有，Logistic应用广泛的原因是许多现实问题跟它的模型吻合。例如一件事情是否发生跟其他数值型自变量的关系。

注意：如果自变量为字符型，就需要进行重新编码。一般如果自变量有三个水平就非常难对付，所以，如果自变量有更多水平就太复杂。这里只讨论自变量只有三个水平。非常麻烦，需要再设二个新变量。共有三个变量，第一个变量编码1为高水平，其他水平为0。第二个变量编码1为中间水平，0为其他水平。第三个变量，所有水平都为0。实在是麻烦，而且不容易理解。最好不要这样做，也就是，最好自变量都为连续变量。

**逻辑回归算法优缺点：**

**优点：**

1 实现简单，广泛的应用于工业问题上；

2 分类时计算量非常小，速度很快，存储资源低；

3 便利的观测样本概率分数；

4 对逻辑回归而言，多重共线性并不是问题，它可以结合L2正则化来解决该问题。

**缺点：**

1 当特征空间很大时，逻辑回归的性能不是很好；

2 容易欠拟合，一般准确度不太高

3 不能很好地处理大量多类特征或变量；

4 只能处理两分类问题（在此基础上衍生出来的softmax可以用于多分类），且必须线性可分；

5 对于非线性特征，需要进行转换。

## 集成方法

TODO随机森林算法 AdaBooting GBDT XGB

## 神经网络算法

TODO ANN CNN RNN

## 图模型算法

TODO

## 文本挖掘

TODO

1. <http://www.crisp-dm.org/>提供的CRISPWP-0800.pdf [↑](#footnote-ref-1)
2. <http://faculty.smu.edu/tfomby/eco5385_eco6380/data/SPSS/SAS%20_%20SEMMA.pdf> [↑](#footnote-ref-2)
3. <http://www.kdd.org/> [↑](#footnote-ref-3)
4. <http://www2.cs.uregina.ca/~dbd/cs831/notes/kdd/1_kdd.html> [↑](#footnote-ref-4)
5. <http://recipp.ipp.pt/bitstream/10400.22/136/1/KDD-CRISP-SEMMA.pdf> [↑](#footnote-ref-5)
6. Höppner, F. Fuzzy cluster analysis: methods for classification, data analysis and image recognition[M]. John Wiley & Sons, 1999. [↑](#footnote-ref-6)
7. https://zh.wikipedia.org/wiki/%E5%86%B3%E7%AD%96%E6%A0%91 [↑](#footnote-ref-7)