本课时，我将为你介绍数据挖掘操作流程的第三个环节，模型训练。

在上一课时，我们解决了一系列又脏又累的数据问题，现在终于可以进入模型训练阶段了。

在数据挖掘中，算法是很多的，而且随着大家研究的深入，有越来越多的优秀算法被设计出来。所以，该怎么去选一个适合需求的算法呢？首先你得明白你面对的是什么问题，虽然算法众多，但是要解决的难题往往有共同点，针对每一类型的问题，就可以找到对应的算法，再根据算法的特性去进行选择。这一课时，我就来介绍一下在工作中最常遇到的一些问题。

分类问题

在内容理解场景下，我**遇到最多的问题**就是分类问题。一个用户写了一篇游记，我想对它做非常详细的理解，比如：

- **游记类别**，是关于 “滑雪” 的，还是关于 “徒步” 的；
- **用户情感**，是 “正向” 的，还是 “负向” 的；
- **内容质量**，是 “高”“中”，还是 “低”；
- **内容风险**，是 “有风险” 的，还是 “无风险” 的。

诸如此类给数据进行明确标签区分的问题都可以看作分类问题。

分类是有监督的学习过程。 处理分类问题首先要有一批已经有标签结果的数据，经过分类算法的学习，就可以预测新的未知数据的分类。如果缺少这些已知的信息，那分类就没办法进行，要么考虑使用其他方法，如聚类算法，要么考虑处理数据，比如说人工进行标注。

分类问题中包括以下 3 种情况：

- **二分类。** 这是分类问题里最简单的一种，因为要回答的问题只有 “是” 或“否”。比如我在处理用户内容时，首先要做一个较大的分类判断，即一条内容是否属于旅游相关内容，这就是二分类问题，得出的结论是这条内容要么是旅游相关，要么不是旅游相关。
- **多分类。** 在二分类的基础上，将标签可选范围扩大。要给一条内容标注它的玩法，那种类就多了，比如冲浪、滑雪、自驾、徒步、看展等，其种类甚至多达成百上千个标签。
- **多标签分类。** 是在多分类基础上再升级的方法。对于二分类和多分类，一条内容最后的结果只有一个，标签之间是互斥的关系。但是多标签分类下的一条数据可以被标注上多个标签。比如一个人在游记里既可以写玩法，也可以写美食，这两者并不冲突。

由于分类问题众多，所以用来解决分类问题的算法也非常多，像 KNN 算法、决策树算法、随机森林、SVM 等都是为解决分类问题设计的。看到这些名字你可能会感到陌生，但是不要担心，关于算法的具体细节，我会在后面的课时进行讲解。

聚类问题

跟分类不同，**聚类是无监督的，** 也就是说没有已经标注好的结果数据供算法学习。你只知道一些数据，而且你需要为这些数据分组，甚至很多时候你连要划分多少个组都不清楚。比如，在一个旅游 APP 上有上千万的用户，你可能会需要把用户划分成若干个组，以便针对特定的用户群体去开发一些特定的功能，比如为爱滑雪的用户，推送一些滑雪的信息。但是用户数量很大，用户的属性也很多，这个时候你就要用到聚类分析。

聚类就是把一个数据集划分成多个组的过程，使得组内的数据尽量高度集中，而和其他组的数据之间尽量远离。这种方法是针对已有的数据进行划分，不涉及未知的数据。

既然是要划分小组，就要先看看小组之间可能存在的 4 种情况。

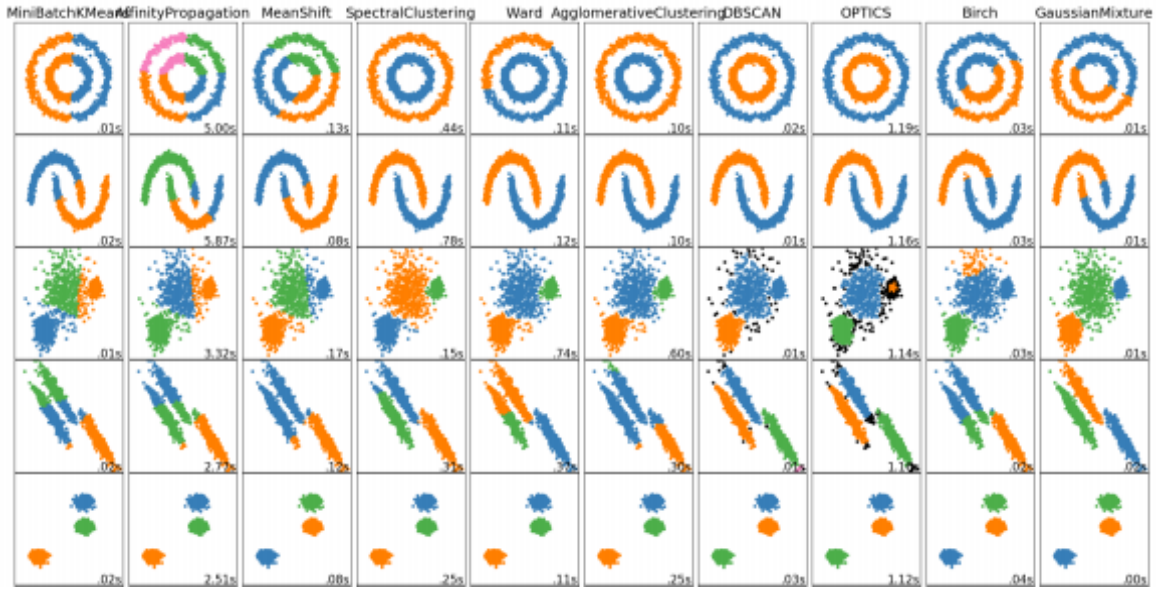
- **互斥**：小组和小组之间是没有交集的，也就是说一个用户只存在于一个小组中。

- **相交**：小组和小组之间有交集，那么一条数据可能既存在于 A 组，也存在于 B 组之中，如一个用户既可以爱滑雪，也可以爱爬山。
- **层次**：一个大组还可以细分成若干个小组，比如将高消费用户继续细分，可以有累积高消费用户和单次高消费用户。
- **模糊**：一个用户并不绝对属于某个小组，只是用概率来表示他和某个小组的关系。假设有五个小组，那么他属于这五个小组的模糊关系就是 [0.5,0.5,0.4,0.2,0.7]。

所以，对应上面 4 种不同的小组情况，也有 4 种不同的聚类方法。

- **第一种：基于划分的聚类，通常用于互斥的小组。** 划分的方法就好像在数据之间画几条线，把数据分成几个小组。想象你的数据散落在一个二维平面上，你要把数据划分成三个类，那么在划分完之后，所有数据都会属于一个类别。
- **第二种：基于密度的聚类，可以用来解决数据形状不均匀的情况。** 有些数据集分布并不均匀，而是呈现不规则的形状，而且组和组之间有一片空白区域，这个时候用划分的方法就很难处理，但是基于密度的聚类不会受到分布形状的影响，只是根据数据的紧密程度去聚类。
- **第三种：基于层级的聚类，适用于需要对数据细分的情况。** 就像前面说的要把数据按照层次进行分组，可以使用自顶向下的方法，使得全部数据只有一个组，然后再分裂成更小的组，直到满足你的要求。如有从属关系，需要细分的数据，就非常适合这种方法。同样，也可以使用自底向上的方法，最开始每一条数据都是一个组，然后把离得近的组合并起来，直到满足条件。
- **最后一种：基于模型的聚类。** 这种聚类方法首先假设我们的数据符合某种概率分布模型，比如说高斯分布或者正态分布， 那么对于每一种类别都会有一个分布曲线，然后按照这个概率分布对数据进行聚类，从而获得模糊聚类的关系。

下面是我在 scikit-learn 官网上找的一张聚类算法对比图，通过数据图形化的形式，对比了针对不同类型的数据情况，使用各个聚类算法得到的结果。图中的每一行都是同一种数据通过不同算法得出的聚类结果，不同颜色表示使用了某个聚类算法之后，该数据集被聚成的类别情况，可以比较直观地理解某个算法适合什么样的数据集。比如第一行的数据分布是两个圆环，可以看到第四个、第六个、第七个和第八个算法的效果比较好，能够成功地按照两个圆环去聚成类别，而其他几种的效果就比较差了。



聚类算法对比图

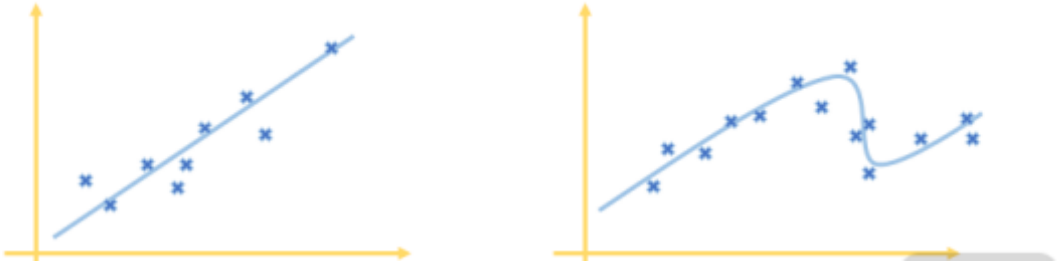
回归问题

与分类问题十分相似，都是根据已知的数据去学习，然后为新的数据进行预测。但是不同的是，分类方法输出的是**离散的标签**，回归方法输出的结果是**连续值**。

那什么是离散的标签呢？“云青青欲雨”这个典型的分类问题，就能说明这一问题。根据特征 “云青青” 输出“雨”，雨、雪、阴、晴这类标签，就是不连续的。而回归就是要通过拟合数据找到一个函数，当你有一组新的数据时，就可以根据这个函数算出一个新的结果值。

回归的英文单词是 Regression，有消退、复原的含义，这种方法是由生物统计学家高尔顿发明的。他在统计父母身高与子女身高的关系时发现，父母的身高都非常高或者都非常矮的情况下，子女身高经常出现衰退的情况，也就是高的父母孩子变矮，矮的父母孩子变高，有回归到平均身高的倾向。

我画了两幅图来说明回归的结果，就是要找到一条线：



回归分析图

这张图不是要进行对比，只是想告诉你不管是线性还是非线性的数据，都可以使用回归分析。

可以看到，数据散落在坐标系上，通过学习你可以得到一条线，较好地拟合了这些数据。这条线可能不通过任何一个数据点，而是使得所有数据点到这条线的距离都是最短的，或者说是损失最小的。根据这条线，如果给出一个新的 x，那么你就能算出对应的 y 是多少。

事实上，回归方法和分类方法可以**相互转化**，比如：

- 在使用回归方法得到函数方程式以后，你可以根据对新数据运算的结果进行区间分段，高于某个阈值给定一个标签，低于该阈值给定另外一个标签。比如你使用回归方法预测完房价之后，不想让客户看到真实的房价，而是给予一个范围的感受，就可以设定高于 500w 的就是 “高房价” 标签，低于 100w 的就是 “低房价” 标签；
- 相反，对于通过分类方法得到的标签，你可以根据给定标签的概率值为其增加一些运算逻辑，将标签转换到一个连续值的结果上。

下面我再用一个表格总结一下分类和回归的情况，你在实际使用时可以根据自己的需要进行选择和处理。

	分类	**回归**
输出	离散数据	连续数据
目的	寻找决策边界	找到最优拟合

分类问题和回归问题总结表

关联问题

关联问题对应的方法就是关联分析。这是一种**无监督学习**，它的目标是挖掘隐藏在数据中的关联模式并加以利用。与分类和回归不同，**关联分析是要在已有的数据中寻找出数据的相关关系**，以期望能够使用这些规则去提升效率和业绩。比如在我们津津乐道的啤酒与尿布的故事中，通过分析销售产品的情况发现，很多购买尿不湿的人也会去买啤酒。你可能不知道这到底是出于什么原因，但是这背后却隐藏着巨大的经济效益，这个案例我会在后面的课时具体分析。

所以，关联分析被广泛地用于各种商品销售分析、相关推荐系统分析、用户行为分析等情况。比如我在第一课时中举的京东套装推荐的例子，就是利用这样的挖掘方法得到的结果，给用户进行各种套装推荐、搭配商品销售、特价组合等。但是在进行大量数据的关联分析时，你会发现各种奇怪的组合，这可能是数据偏差产生的影响，所以在最终结果应用的时候还需要加入一些知识校验。

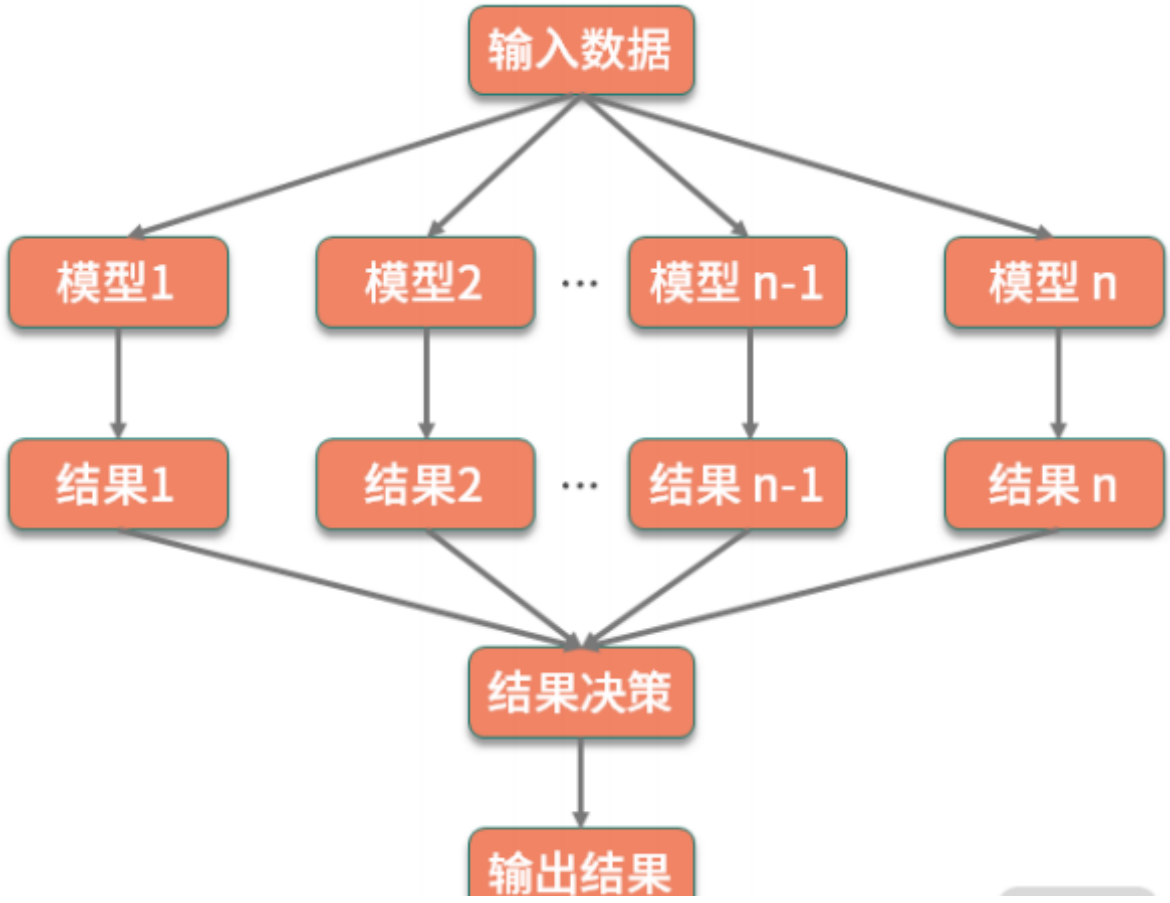
说到这里，我已经把四大问题都介绍完了，每个问题都可以通过相应的机器学习算法来进行解决。但是，在实践的时候，很多问题不是靠一个算法、一个模型就能解决的，往往要针对具体的细节使用多个模型以获得最佳效果，所以就要用到模型集成。

模型集成

模型集成也可以叫作集成学习，其思路就是去合并多个模型来提升整体的效果。

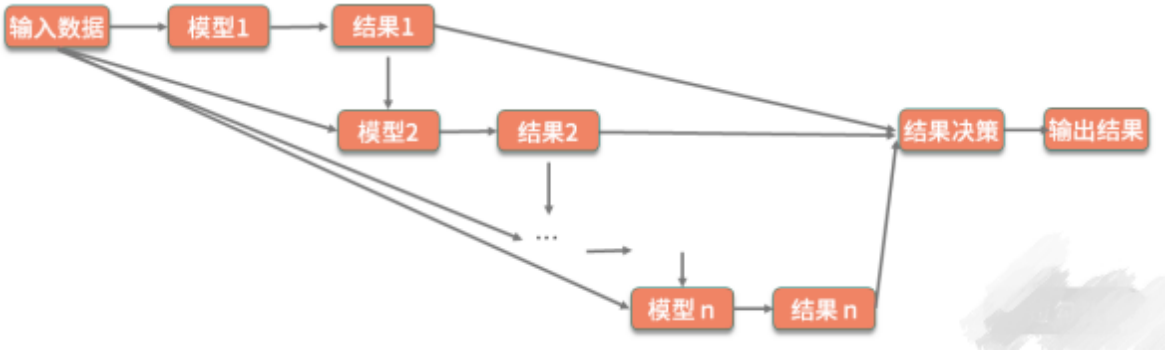
既然是要合并多个模型，那么很容易想到训练多个并列的模型，或者串行地训练多个模型。下面我就来讲一下模型集成的 3 种方式。

- **Bagging（装袋法）：** 比如多次随机抽样构建训练集，每构建一次，就训练一个模型，最后对多个模型的结果附加一层决策，使用平均结果作为最终结果。随机森林算法就运用了该方法，这种算法我会在后面的课时具体介绍。这一方法的过程如下图所示：



装袋法图

- **Boosting（增强法）：** 这个就是串行的训练，即每次把上一次训练的结果也作为一个特征，不断地强化学习的效果。



增强法图

- **Stacking（堆叠法）：** 这个方法比较宽泛，它对前面两种方法进行了扩展，训练的多个模型既可以进行横向扩展，也可以进行串行增强，最终再使用分类或者回归的方法把前面模型的结果进行整合。其中的每一个模型可以使用不同的算法，对于结构也没有特定的规则，真正是 “黑猫白猫，抓住老鼠就是好猫”。所以，在使用堆叠法时，就需要你在具体业务场景中不断地去进行尝试和优化，以达到最佳效果。

总结

在这一课时，我介绍了工作中最常见的四大问题以及模型集成，我想你应该学到了这些问题的内部机理，并且知道要解决这些问题需要有什么样的思路。但是在这一课时中，我并没有介绍算法的细节，别担心，我会在后面的课时中详细展开。

你可以思考一下，在平时的工作生活中，除了这四种问题是不是还有别的问题可以用到数据挖掘来解决呢？你遇到的问题是否可以通过相互转化变成这 4 种问题中的一种来进行处理呢？欢迎将你的思考和疑问写在留言区，与我和其他同学分享交流。