第5章

# 信息论与熵

现有的人工智能技术与相对成熟的信息论有着千丝万缕的联系。例如，特征抽取与选择其实就是剔除冗余信息，对数据进行“压缩编码”，从而实现其高效表示的一种方式。从训练集所有数据的“编码结果”中发现规律其实就是一种解码过程。需要说明的是，信息论的发展本质是为了提升数据传输的质量与速度。而人工智能处理技术面对的是从已有数据中发现规律，并用于实现对新数据类别或评估结果的正确预测。因此，与信息论中解码器实现数据高质量恢复不同，人工智能算法的解码过程关注隐藏在数据背后的规律的显式表达。除此之外，给定预测集 $X=\{\boldsymbol{x}_1,\boldsymbol{x}_2,\cdots,\boldsymbol{x}_N\}$，设其对应的真实值变量 $\hat{Y}$ 服从某一分布 $\hat{\Gamma}$。数据驱动的人工智能算法的本质是使得预测值变量 $Y$ 的分布函数 $\Gamma$ 与真实分布 $\hat{\Gamma}$ 尽可能相等。这一目标通常由最小化交叉熵损失函数来实现。本章重点介绍人工智能领域涉及的信息论知识。

## 5.1 人工智能与信息论

有人说“人工智能与信息论构成一枚硬币的正反两面”。实际上，发展更加成熟的信息论为人工智能技术的成长提供了许多营养供给。例如，数据驱动的 Encoder-Decoder 模型本质上是一个单工通信系统。这一点从其名字中即可发现一些蛛丝马迹。不同的是，通信系统中的编码与解码重点关注的是信息传播的便捷性，即便存在编码损失、传输干扰，其解码结果通常是对原数据的“保真恢复”。而人工智能技术更关注海量数据中对待解决问题有指导意义信息的有效表达（编码），以及从有价值信息中推演（解码）待解决问题答案的问题。也就是说，通常地，人工智能领域中的解码技术并不是为了完全恢复编码前的数据，而是从中发现蕴藏其中的与待解决问题相关的规律、模式。例如，给定句子对 $\langle X,Y\rangle$，自然语言翻译领域的 Encoder-Decoder 模型可实现由语句 $X$ 到语句 $Y$ 的翻译工作。具体地，若语句 $X$ 由 $n$ 个词组构成，设第 $i$ 个词组的嵌入表示为 $x_i$，则 $X=\{\boldsymbol{x}_1,\boldsymbol{x}_2,\cdots,\boldsymbol{x}_n\}$，其中，$i=1,2,\cdots,n$。与之对应地，语句 $Y$ 由 $m$ 个词组构成，第 $j$ 个词组的嵌入表示为 $\boldsymbol{y}_j$，则 $Y=\{\boldsymbol{y}_1,\boldsymbol{y}_2,\cdots,\boldsymbol{y}_m\}$，其中，$j=1,2,\cdots,m$。需要说明的是，在自然翻译领域，词组的嵌入表示通常指的是该词组

的特征向量表示。另外，构成一个句子的词组的嵌入表达是有序的。在以上形式化表达的基础上，编码任务即是找到语句 $X$ 的中间语义表示 $\boldsymbol{S}$，即 $\boldsymbol{S}=\mathcal{E}(\boldsymbol{x}_1,\boldsymbol{x}_2,\cdots,\boldsymbol{x}_n)$。与之对应地，对于解码器来说，其任务是根据语句 $X$ 的中间语义表示 $\boldsymbol{S}$ 和之前已翻译完成的 $k$ 个词组 $\{\tilde{\boldsymbol{y}}_1,\tilde{\boldsymbol{y}}_2,\cdots,\tilde{\boldsymbol{y}}_k\}$，来重构第 $k+1$ 个词组 $\tilde{\boldsymbol{y}}_{k+1}$，其中，$1\leqslant k<m$。也即，$\tilde{\boldsymbol{y}}_{k+1}=\mathcal{D}(\boldsymbol{S};\tilde{\boldsymbol{y}}_1,\tilde{\boldsymbol{y}}_2,\cdots,\tilde{\boldsymbol{y}}_k)$。其中，$k=1,2,\cdots,m-1$。

其实，从不确定性较高的海量数据中得到确定性答案的过程，是一个信息不确定度不断下降的过程。在信息论中，信息的不确定性是由信息熵来表达的。信息熵在决策树分类、特征选择、损失函数构建等方面均起着重要作用。本章接下来将就人工智能领域中的编码、解码、熵相关信息论知识展开详细介绍。

注

有必要指出的是，实际上，人工智能与信息论两个学科互相交叉。但主要还是人工智能借用更加成熟的信息论方法，用于推进其理论研究进展，拓展其应用场景。一个比较典型的例子就是借鉴信息理论创造和改进学习算法，甚至已经衍生出一个称作信息理论学习的研究方向。

## 5.2 特征编码

数据驱动的人工智能方法通常由原始数据集 $D$ 中抽取特征向量，并组成特征矩阵 $F$，即寻找映射 $f:D\to F$。这一过程可由人工设计完成，也可由包括神经网络在内的智能模型来生成。由特征矩阵 $F$ 来表示待分析数据 $D$ 本质上就是实现对 $D$ 的有效编码。具体地，每种特征的结构、取值范围、长度，以及特征与特征之间的组合方式均是信息编码的体现。

### 5.2.1 直接编码

设一特征向量由体重与性别两种特征值构成。不难理解，由于体重取连续浮点值，其测量结果可直接写入特征向量。不考虑生物学上的特殊情况，性别特征取值有“男”与“女”两种。也就是说，一个二进制位的两种取值状态足以实现对性别特征的编码表达。类似地，若某一特征由 $N$ 个不同状态取值构成，则其取值可用 $\{0,1,\cdots,N-1\}$ 表示。需要说明的是，若存在取值未知特征，则需要多增加一个状态，用于表示缺失值。若一个特征向量由性别、归属地、浏览器工具三类特征构成，且性别特征 $S$ 取值空间为{男，女}，归属地 $L$ 取值空间为{亚洲，欧洲，非洲，美洲}，浏览器工具 $B$ 取值空间为{Chrome，IE，360，Firefox，Safari}，则以上三类特征可直接编码表示为 $S=\{0,1\}$，$L=\{0,1,2,3\}$，$B=\{0,1,2,3,4\}$。不难理解，对于样本“来自欧洲使用 Firefox 的男性”来说，其特征向量的一种直接编码结果为[0,1,3]。需要说明的是，实际上 0,1,3 的任意“拼接”均可用于表示以上样本，主要取决于特征向量中不同特征所处位置的差异性。这本身也是特征编码的一种体现。

注

并不是连续值型变量一定不需要重新编码。特别地，在有些应用场景下，连续型变量需要离散化处理以解决实际问题。

### 5.2.2　One-hot 编码

又称一位有效编码。该编码方法用一个二进制位独立表达特征的某一取值状态，并且编码结果中有且仅有一个二进制位是有效位。也就是说，对于包含 4 种取值的某特征，直接编码采用 2 个比特位来表示其各个取值状态。而 One-hot 编码采用 4 个比特位来表示，每个比特位与一个状态对应。也就是说，一位有效编码码字总位数取决于特征状态数，每一个码字里“1”的位置，代表对应状态生效。例如，包含 4 种取值特征编码依次为 0001、0010、0100、1000。若一个特征向量由性别、归属地、浏览器工具三类特征构成，且性别特征 S 取值空间为{男，女}，归属地 $L$ 取值空间为{亚洲，欧洲，非洲，美洲}，浏览器工具 $B$ 取值空间为{Chrome，IE，360，Firefox，Safari}，则以上三类特征的 One-hot 编码可分别对应表示为 $S=\{01,10\}$，$L=\{0001,0010,0100,1000\}$，$B=\{00001,00010,00100,01000,10000\}$。不难理解，对于样本“来自欧洲使用 Firefox 的男性”来说，其特征向量的 One-hot 编码结果为[01，00，0000，0010，0000，0000，00000，00000，00000，01000，00000]，可进一步简写为[1，0，0，1，0，0，0，0，0，1，0]。

### 5.2.3　Dummy 编码

Dummy 编码又称哑变量编码、虚拟变量编码。不难发现，One-hot 编码保证每个取值只有一种状态被“激活”，即只有一个状态位取值为 1，其他状态位均被设置为 0。实际上，若一个离散型特征变量只有 $N$ 种不同取值状态，则用 $N-1$ 位即可表达其所有状态取值情况。这是因为，例如，若归属地 $L$ 取值空间为{亚洲，欧洲，非洲，美洲}，则不是亚洲、欧洲、非洲的归属地，必然是美洲。也就是说，若前三个归属地依次编码为 0001，0010，0100，则全 0 状态位 0000 即可代表美洲。以上编码可进一步化简为 001，010，100，000。

注

有必要说明的是连续型变量的离散化处理，如有效编码与哑变量编码，均可提升模型的非线性能力。这是因为，就线性模型来说，给定连续型变量 $x$，其权重控制参数只有一个，设为 $w$；而离散化处理后，其对应取值为 $N$ 个，记作 $x_1,x_2,\cdots,x_n$，权重控制参数也由原来的一个增加为 $N$ 个。显然，这使得对变量 $x$ 的控制参数管理更加精细。这显然可以提升原模型的非线性表达能力。除此之外，与由一个很大的权值管理一个特征相比，拆分成众多小权值管理这个特征，可降低特征值扰动、异常数据对模型稳定性的影响，进而使得模型鲁棒性更强。

## 5.3 压缩编码

信息论中编码的主要任务是尽可能保证信息完整性的前提下，降低其对传输带宽或存储空间的占用。也就是说，其背后均蕴含着对数据的压缩处理思想。这一点与数据驱动的人工智能领域涉及的多项技术紧密相关。

### 5.3.1 聚类

聚类的目的通常是将数据集中样本划分为若干互不相交的子集，使得每个子集与一个现实中的概念相对应，并用子集“中心”作为对应概念的抽象表达。形式化地，给定包含 $m$ 个样本的数据集 $D=\{\boldsymbol{x}_1,\boldsymbol{x}_2,\cdots,\boldsymbol{x}_m\}$，其中每个样本均由一个 $n$ 维特征向量来表示，即 $\boldsymbol{x}_i=[x_{i,1},x_{i,2},\cdots,x_{i,n}]$，其中 $i=1,2,\cdots,m$。设某一聚类算法将样本数据集 $D$ 划分为 $k$ 个互不相交子集，分别记作 $D_1,D_2,\cdots,D_k$，其中，$D=\bigcup_{j=1}^{k}D_j$ 且 $D_j\cap D_{l\neq j}=\varnothing$。若 $k$ 个子集对应语义分别记作 $S_j$，其中，$j=1,2,\cdots,k$，则聚类的实质是在样本数据集 $D$ 与语义概念之间建立映射 $F$。也就是说，$F$ 实现了样本数据集 $D$ 的语义压缩编码。

**注**

数据驱动的人工智能领域存在许多聚类方法，大致可划分为 $K$ 均值聚类、模糊 $C$ 均值聚类、高斯混合聚类、密度聚类、谱聚类等。感兴趣的读者可查阅相关资料。

### 5.3.2 特征降维

在数据驱动的人工智能领域，用于解决实际问题的方法是否有效，与样本数据的特征表达强相关。不幸的是，特征向量维度的增加，导致样本数据在其空间内的分布密度急剧降低。与此同时，维度的增加必然给样本间距离的度量带来额外开销。这在人工智能领域被称作维度灾难。解决这一问题的有效途径之一，是特征降维。这是因为，很多时候虽然特征向量分布在高维空间，但是待求解问题的解只与这一高维空间中的一个低维嵌入相关。显然，随机剔除组成整个特征向量的部分特征值，可以起到缓解维度灾难的作用。但这不可避免地导致有价值信息的丢失。因此，一类特征降维方法基于降低特征维度的同时，确保高维空间中样本距离在低维空间中保持不变的前提。

形式化地，给定包含 $m$ 个样本的数据集 $S=\{\boldsymbol{x}_1,\boldsymbol{x}_2,\cdots,\boldsymbol{x}_m\}$，其中每个样本均由一个 $n$ 维特征向量来表示，即 $\boldsymbol{x}_i=[x_{i,1},x_{i,2},\cdots,x_{i,n}]$，其中，$i=1,2,\cdots,m$。所有样本构成特征矩阵 $\boldsymbol{A}\in R^{m\times n}$，则样本间距离在 $n$ 维特征空间内构成 $m$ 阶距离矩阵 $\boldsymbol{D}$。其中，第 $i$ 行 $j$ 列元素为样本 $\boldsymbol{x}_i$ 与样本 $\boldsymbol{x}_j$ 之间的欧氏距离，记作 $D_{i,j}=d_{\text{欧}}(\boldsymbol{x}_i,\boldsymbol{x}_j)=\|\boldsymbol{x}_i-\boldsymbol{x}_j\|$。若降维后样本特征向量长度由 $n$ 维缩减为 $d$ 维，其中，$d\leqslant n$，则样本新特征向量构成特征矩阵 $\boldsymbol{Z}\in R^{m\times d}$。样本 $\boldsymbol{x}_i$ 与 $\boldsymbol{x}_j$ 在 $d$ 维空间中与特征矩阵 $\boldsymbol{Z}$ 中第 $i$ 行、第 $j$ 行元素 $\boldsymbol{z}_i$ 与 $\boldsymbol{z}_j$ 一一对应。为保持距离不变性，也就是说，$D_{i,j}=\|\boldsymbol{z}_i-\boldsymbol{z}_j\|$ 恒成立。与距离矩阵 $\boldsymbol{D}$ 类似，构造降维后样本特征向量间内积矩阵 $\boldsymbol{B}$，其中，$B_{i,j}=\boldsymbol{z}_i\boldsymbol{z}_j^{\mathrm{T}}$，显然其为 $m$ 阶矩阵。也就是说，

内积矩阵 $\boldsymbol{B}$ 与特征矩阵 $\boldsymbol{Z}$ 之间存在如下关系：$\boldsymbol{B}=\boldsymbol{Z}\boldsymbol{Z}^{\mathrm{T}}$。由 $D_{i,j}=\|\boldsymbol{z}_i-\boldsymbol{z}_j\|$，可得

$$\begin{aligned} D_{i,j}^2 &= \|\boldsymbol{z}_i\|^2+\|\boldsymbol{z}_j\|^2-2\boldsymbol{z}_i\boldsymbol{z}_j^{\mathrm{T}} \\ &= B_{i,i}+B_{j,j}-2B_{i,j} \end{aligned} \tag{5-1}$$

设降维后样本已中心化处理，即 $\sum_{i=1}^{m}\boldsymbol{z}_i=0$。易得

$$\sum_{i=1}^{m}B_{i,j}=\sum_{i=1}^{m}\boldsymbol{z}_i\boldsymbol{z}_j^{\mathrm{T}}=\left(\sum_{i=1}^{m}\boldsymbol{z}_i\right)\boldsymbol{z}_j^{\mathrm{T}}=0 \tag{5-2}$$

类似地，$\sum_{j=1}^{m}B_{i,j}=0$。

> **注**　中心化的作用与意义详见第 7 章。另外，降维后样本中心化处理后，内积矩阵实际应对样本的协方差矩阵。

结合以上各式，可得

$$\sum_{i=1}^{m}D_{i,j}^2=\mathrm{tr}(\boldsymbol{B})+mB_{j,j} \tag{5-3}$$

$$\sum_{j=1}^{m}D_{i,j}^2=\mathrm{tr}(\boldsymbol{B})+mB_{i,i} \tag{5-4}$$

$$\sum_{i=1}^{m}\sum_{j=1}^{m}D_{i,j}^2=2m\,\mathrm{tr}(\boldsymbol{B}) \tag{5-5}$$

其中，$\mathrm{tr}(\boldsymbol{B})$表示矩阵 $\boldsymbol{B}$ 的迹。结合式(5-1)，易得

$$\begin{aligned} B_{i,j} &= -\frac{1}{2}(D_{i,j}^2-B_{i,i}-B_{j,j}) \\ &= -\frac{1}{2}\left(D_{i,j}^2-\frac{1}{m}\sum_{i=1}^{m}D_{i,j}^2-\frac{1}{m}\sum_{j=1}^{m}D_{i,j}^2+\frac{1}{m^2}\sum_{i=1}^{m}\sum_{j=1}^{m}D_{i,j}^2\right) \end{aligned} \tag{5-6}$$

也就是说，降维后的任意样本对之间的内积可由降维前二者间的欧几里得距离唯一表示。由 1.16.2 节可知，只需对内积矩阵 $\boldsymbol{B}$ 进行特征值分解便可以得到特征矩阵 $\boldsymbol{Z}$。不难理解，降维后的特征向量可视作对原始特征向量的压缩编码。

### 5.3.3　特征选择

特征降维是解决“维度灾难”的有效途径之一。若特征集中部分特征对待解决问题的求解有指导性意义，则称为相关特征，否则称为无关特征。由给定特征集选择出相关特征子集，有助于降维或降低待解决问题的难度。需要说明的是，特征降维关注的是特征向量的长度。降维后的向量中任意元素段均可能失去原有语义。而特征选择更关注特征类别的取舍，被选中特征的完整性并未被破坏。特征选择必须保证不丢失对待解问题求解重要的特征。待求解问题不同，相关特征也不尽相同。

有必要指出的是，给定特征集合，若没有任何先验知识，遍历所有的可能子集不可避免地导致组合爆炸问题。一种可能的解决方法是，构造候选子集，对其进行评价，并由评

价结论修正候选子集。持续以上步骤直到无法找到更优的候选子集为止。形式化地，给定特征集合 $A=\{a_1,a_2,\cdots,a_d\}$，最小候选集只有一个特征构成。也就是说，首先对特征集合中的 $d$ 个特征依次进行评价。假设 $a_3$ 最优，则 $A_1=\{a_3\}$ 为当前最优候选子集。然后将除特征 $a_3$ 之外的其他 $d-1$ 个特征与 $a_3$ 依次构成包含两个特征元素的候选子集。假设 $A_2=\{a_3,a_1\}$ 最优，且优于 $\{a_3\}$，则将 $\{a_3,a_1\}$ 作为本轮候选特征子集。直到第 $k+1$ 轮时拥有 $k+1$ 个特征最优候选子集 $A_{k+1}$ 的评价结果均不如第 $k$ 轮生成的拥有 $k$ 个特征的最优候选子集 $A_k$，则 $A_k$ 为最终的特征选择结果。

**注**

除采取逐轮搜索增加相关特征的最优候选子集外，还可由全集出发，采用逐轮减少无关特征数量的搜索策略。或者将二者有机结合。另外需要指出的是，上述搜索策略均是基于贪心机制的，可能陷入局部最优。

由以上描述不难发现，最优候选子集的构造过程离不开对其优化程度的评价。信息增益是这类评价中的典型代表。给定特征子集 $B$，假定根据其取值可将样本数据集 $D$ 划分为 $v$ 个子集，即 $\{D_1,D_2,\cdots,D_v\}$，其中每个子集中的样本在特征子集 $B$ 上取值相同，则特征子集 $B$ 的信息增益定义为

$$G(B)=\mathrm{Ent}(D)-\sum_{i=1}^{v}\frac{\mathrm{card}(D_i)}{\mathrm{card}(D)}\mathrm{Ent}(D_i) \tag{5-7}$$

其中，$\mathrm{Ent}(D)$ 与 $\mathrm{Ent}(D_i)$ 为样本数据集 $D$ 及其子集 $D_i$ 的信息熵。

**注**

信息熵与各类样本占总体的比值有关。熵值越大表示数据类别多样性越强。反之，数据类别单一，纯度更高。因此，若特征子集使得式(5-7)定义的信息增益越大，则意味着此候选子集更有利于类别划分。关于信息熵的定义与性质详见 5.7 节。有必要说明的是，以上定义的信息增益只是候选子集评价的一种途径，任何一种可判断划分差异性的评价均可用于最优候选子集筛选。另外，有必要指出的是，特征权重正则化也是特征选择的一种有效策略。关于正则化详见第 7 章。

### 5.3.4 稀疏编码

来看现实生活中的实例，通常图书均有目录，字典有拼音查字法、部首查字法等，这里的目录以及多种不同的查字方法，实际是在海量信息与其代表的核心信息之间建立了一个快速定位索引。换个角度，给定一份文档以及一本字典，则该文档中同一汉字出现的频次构成一个特征向量。此特征向量的长度为字典中包含汉语的个数。通常地，受主题及上下文环境影响，一份文档不可能涵盖字典中所有的汉字。也就是说，生成的特征向量中包含多个零元，具备稀疏性。不难理解，维度相同时，稀疏分布的样本更易区分。构造字典使得特征表达具备稀疏性是数据驱动的人工智能解决实际问题的一种有效策略。

形式化地，给定样本数据集 $D=\{\boldsymbol{x}_1,\boldsymbol{x}_2,\cdots,\boldsymbol{x}_m\}$，其中每个样本均由一个 $n$ 维特征向

量来表示，即 $\boldsymbol{x}_i=[x_{i,1},x_{i,2},\cdots,x_{i,n}]$，其中，$i=1,2,\cdots,n$。所有样本构成特征矩阵 $\boldsymbol{A}\in R^{m\times n}$。构造词汇量规模为 $k$ 的字典矩阵 $\boldsymbol{B}\in R^{k\times n}$，使得任意特征向量 $\boldsymbol{x}_i=[x_{i,1},x_{i,2},\cdots,x_{i,n}]$ 均与其编码后的特征向量 $\boldsymbol{y}_i=[y_{i,1},y_{i,2},\cdots,y_{i,k}]$ 保持线性关系，即 $\boldsymbol{x}_i=\boldsymbol{y}_i\boldsymbol{B}$。为求得字典矩阵 $\boldsymbol{B}$ 及稀疏表达的特征向量集 $\{\boldsymbol{y}_1,\boldsymbol{y}_2,\cdots,\boldsymbol{y}_m\}$，最小化如下目标函数：

$$\min_{B,y_i}\sum_{i=1}^{m}\left(\left\|\boldsymbol{x}_i-\boldsymbol{y}_i\boldsymbol{B}\right\|_2^2+\lambda\left\|y_i\right\|_1\right) \tag{5-8}$$

其中，上式第二项编码后特征向量的 1-范数，起正则化作用。关于范数与正则化详见第 7 章。

> **注**
>
> 有必要说明的是，物极必反。当稀疏到一程度时，再增加样本特征表达的稀疏性可能给待解决问题的求解带来新的挑战。

### 5.3.5　压缩感知

给定样本的 $n$ 维特征向量 $x$，考虑线性变换构成矩阵 $\boldsymbol{\Phi}\in R^{n\times m}$，且满足 $\boldsymbol{y}=\boldsymbol{x}\boldsymbol{\Phi}$，显然变换结果 $y$ 为 $m$ 维向量，其中，$m<<n$。也就是说，线性变换矩阵 $\boldsymbol{\Phi}$ 实现了特征向量的压缩编码。但是，给定变换结果 $m$ 维特征向量 $y$ 以及变换矩阵 $\boldsymbol{\Phi}$，几乎不可能恢复原特征向量 $\boldsymbol{x}$。这是因为，由于 $m<<n$，所以 $\boldsymbol{y}=\boldsymbol{x}\boldsymbol{\Phi}$ 构成欠定方程组，即方程个数小于变量个数。即便少数变量相互间具备线性关系，方程组解仍不唯一。幸运的是，假设存在另一个线性变换 $\boldsymbol{\Psi}\in R^{n\times n}$，使得 $\boldsymbol{x}=\boldsymbol{s}\boldsymbol{\Psi}$。显然，$s$ 也是一个 $n$ 维特征向量。也就是说，$\boldsymbol{y}=\boldsymbol{x}\boldsymbol{\Phi}$ 可改写为 $\boldsymbol{y}=\boldsymbol{s}\boldsymbol{\Psi}\boldsymbol{\Phi}$。令 $\boldsymbol{A}=\boldsymbol{\Psi}\boldsymbol{\Phi}$，则可得出如下结论：若能根据特征向量 $\boldsymbol{y}$ 恢复出特征向量 $\boldsymbol{s}$，则可依据 $\boldsymbol{x}=\boldsymbol{s}\boldsymbol{\Psi}$ 进一步恢复出原特征向量 $\boldsymbol{x}$。有必要说明的是，乍看起来，由特征向量 $\boldsymbol{y}$ 恢复出特征向量 $\boldsymbol{s}$ 仍然是一个由 $m$ 维特征向量恢复 $n$ 维特征向量的问题，且变换矩阵 $\boldsymbol{A}$ 仍然是 $n\times m$ 的。也就是说，这仍然是一个求解欠定方程组的问题。有趣的是，若特征向量 $\boldsymbol{s}$ 是稀疏的，即多个位置的特征值为 0，则 $\boldsymbol{s}\boldsymbol{A}$ 构成欠定方程组的可能性减小，有利于上述问题的求解。

不难理解，上述变换矩阵 $\boldsymbol{A}$ 与稀疏编码中字典作用类似，用于将特征向量转换为稀疏表示。但是，与特征选择与稀疏编码不同的是，压缩感知从特征向量本身具备的稀疏性出发，由部分非 0 特征恢复完整向量。幸运的是，通常在原定义域上不具备稀疏性的数据，经过某种线性变换后，在变换域上表现出稀疏性。可能的变换包括傅里叶变换、余弦变换、小波变换等。感兴趣的读者可参阅相关材料。

## 5.4　决策编码

数据驱动的人工智能方法从历史数据中发现规律，实现对未知样本的正确决策。其本质是构造数据与决策结果之间的函数映射。其间通常借助特征向量对数据进行全面、深入、细致的描述。如前文所述，这一数据到特征的转换过程与信息论中数据编码思想相对应。实际上，函数映射的表达形式、求解过程也与信息编码有类似之处。接下来，将详细介绍相关内容。

### 5.4.1 假设空间

对于待求解问题来说，选定模型后，数据驱动的人工智能方法通常可视作参数最优化过程。假设一个模型共有 $m$ 个可变参数，记作 $\theta_1,\theta_2,\cdots,\theta_m$。不难理解，所有参数一起构成一个 $m$ 维假设空间 $\Theta$。而各参数的取值范围则定义了对应维度的界。理想的人工智能方法是在假设空间 $\Theta$ 内找到一个最优参数向量$[\hat{\theta}_1,\hat{\theta}_2,\cdots,\hat{\theta}_m]$。

### 5.4.2 版本空间

不难理解，假设空间通常很大，而训练数据集毕竟有限可数。与欠定线性方程组存在多个可能解类似，数据驱动的人工智能方法很可能由有限训练样本得到多个参数向量。也就是说，存在着多个假设与训练集保持一致。这些假设构成解的“版本空间”。有必要说明的是，构成“版本空间”的多个参数向量对新样本的预测必然存在结论差异风险。这给人工智能决策模型的选择带来不确定因素。这一点与压缩感知中线性变换矩阵的欠定性导致的方程组解不唯一问题类似。解决这一问题的有效策略是引入“归纳偏好”机制，这与压缩感知采用的稀疏性机制有异曲同工之妙。例如，偏好位置的特征值保持不变，其他位置置 0。这一机制的设计灵感来自于人类通常对某大类事物中部分事物具有特殊偏好的事实。需要指出的是，式(5-8)中的 1-范数正则化其实就是对解空间中模较小的参数向量的一种归纳偏好。

> 注
>
> 关于范数与正则化详见第 7 章。

### 5.4.3 决策平面

以线性二分类为例，给定样本数据集 $D=\{\boldsymbol{x}_1,\boldsymbol{x}_2,\cdots,\boldsymbol{x}_m\}$，其中每个样本均由一个 $n$ 维特征向量来表示，即 $\boldsymbol{x}_i=[x_{i,1},x_{i,2},\cdots,x_{i,n}]$，其中，$i=1,2,\cdots,m$。设各样本真实标签依次为 $\hat{y}_1,\hat{y}_2,\cdots,\hat{y}_m$。也就是说，样本数据 $\boldsymbol{x}_i$ 及其真实标签 $\hat{y}_i$ 之间构成数对$\langle \boldsymbol{x}_i,\hat{y}_i\rangle$，其中，$i=1,2,\cdots,m$。另外，实际上真实标签 $\hat{y}_1$ 为二值变量。具体地，若样本数据 $\boldsymbol{x}_i$ 为正例，则 $\hat{y}_i=1$；否则 $\hat{y}_i=0$。线性分类的本质是寻找最优参数向量 $\boldsymbol{w}=[w_1,w_2,\cdots,w_{n+1}]$，使得给定待预测样本 $\boldsymbol{x}=[x_1,x_2,\cdots,x_n]$，当 $\boldsymbol{w}[\boldsymbol{x},1]^{\mathrm{T}}>0$ 时，样本 $\boldsymbol{x}$ 被归类为正例；当 $\boldsymbol{w}[\boldsymbol{x},1]^{\mathrm{T}}<0$ 时，样本 $\boldsymbol{x}$ 被归类为负例。对应地，方程 $\boldsymbol{w}[\boldsymbol{x},1]^{\mathrm{T}}=0$ 即构成了这一问题的决策平面。分类模型确定后，决策面方程中的控制参数值由训练样本数据集 $D=\{\boldsymbol{x}_1,\boldsymbol{x}_2,\cdots,\boldsymbol{x}_m\}$唯一确定。也就是说，就给定的分类问题来说，决策面方程就是对其已知训练样本数据的一种编码表达。

### 5.4.4 纠错输出码

延续上文的分类问题，待解决问题不是二分类任务而是一个多分类任务。也就是说，样本数据 $\boldsymbol{x}_i$ 的真实标签 $\hat{y}_1$ 有多个取值。设类别个数记作 $N$，且真实标签所有取值构成

集合$\{0,1,2,\cdots,N-1\}$。如何借助二分类器完成多分类任务，是数据驱动人工智能研究需要解决的一个问题。有必要指出的是，有些二分类模型可直接推广，用于解决多分类问题，如 KNN、朴素贝叶斯等。更一般地，通常将多分类问题拆解为多个二分类问题，为每个二分类任务训练一个分类器。不难理解，这里的拆分过程与信息论中的编码思想不谋而合。

领域内经典的拆分策略包括一对一、一对其余、多对多三种。对于一对一分类来讲，给定样本数据集 $D=\{x_1,x_2,\cdots,x_m\}$及其真实标签值 $\hat{y}_1,\hat{y}_2,\cdots,\hat{y}_m$，其中 $\hat{y}_i\in\{0,1,2,\cdots,N-1\}$，以类别标签为基准将样本数据划分为 $N$ 个子集。再将各子集两两配对，构成正反例，并分别训练一个二分类器。显然，类别数目为 $N$ 时，一对一策略将引入 $N(N-1)/2$ 个二分类器。这必然导致分类器训练代价大幅上涨。与之对应地，一对其余策略将产生 $N$ 个分类器，开销更小。多对多拆分策略是以上两种策略的范化版本。纠错输出码在该拆分策略中比较常见。该策略将类别与行对应，将分类器与列对应，如图 5-1 所示，若分类器与类别相交处为+1，则表示该分类器将此类别样本视作正例；若为−1 则表示该分类器将此类别样本视作负例；若为 0 则表示该分类器对该类别样本不做识别处理。由图 5-1 不难发现，以上三类拆分策略均可采用纠错输出码来表示。

| | $f_1$ | $f_2$ | $f_3$ | $f_4$ | $f_5$ |
|---|---|---|---|---|---|
| $C_1$ | +1 | −1 | +1 | −1 | −1 |
| $C_2$ | −1 | +1 | −1 | −1 | −1 |
| $C_3$ | −1 | −1 | +1 | −1 | +1 |
| $C_4$ | −1 | +1 | −1 | +1 | +1 |

(a) 多对多

| | $f_1$ | $f_2$ | $f_3$ | $f_4$ |
|---|---|---|---|---|
| $C_1$ | +1 | −1 | −1 | −1 |
| $C_2$ | −1 | +1 | −1 | −1 |
| $C_3$ | −1 | −1 | +1 | −1 |
| $C_4$ | −1 | −1 | −1 | +1 |

(b) 一对其余

| | $f_1$ | $f_2$ | $f_3$ | $f_4$ | $f_5$ | $f_6$ |
|---|---|---|---|---|---|---|
| $C_1$ | +1 | +1 | +1 | 0 | 0 | 0 |
| $C_2$ | −1 | 0 | 0 | +1 | +1 | 0 |
| $C_3$ | 0 | −1 | 0 | −1 | 0 | +1 |
| $C_4$ | 0 | 0 | −1 | 0 | −1 | −1 |

(c) 一对一

图 5-1　纠错输出码示例

## 5.5　决策解码

信息论中编码是为了压缩数据，提升传输性能。接收端需要对数据进行解码恢复操作。虽然人工智能领域的编码、解码应用与其不完全相同，但多数情况下，是需要对编码结果进行类似解码操作的。

### 5.5.1　聚类

给定包含 $m$ 个样本的数据集 $D=\{\boldsymbol{x}_1,\boldsymbol{x}_2,\cdots,\boldsymbol{x}_m\}$，其中每个样本均由一个 $n$ 维特征向量来表示，即 $\boldsymbol{x}_i=[x_{i,1},x_{i,2},\cdots,x_{i,n}]$，其中，$i=1,2,\cdots,m$。设某一聚类算法将样本数据集 $D$ 划分为 $k$ 个互不相交的子集，分别记作 $D_1,D_2,\cdots,D_k$，其中，$D=\bigcup_{j=1}^{k}D_j$ 且 $D_j\bigcap D_{l\neq j}=\varnothing$。设 $k$ 个子集聚类中心，记作 $\boldsymbol{c}_o$。对于预测样本 $\boldsymbol{y}$，若其与各聚类中心的距离构成集合 $S=\{d(\boldsymbol{y},\boldsymbol{c}_1),d(\boldsymbol{y},\boldsymbol{c}_2),\cdots,d(\boldsymbol{y},\boldsymbol{c}_k)\}$，则预测样本 $\boldsymbol{y}$ 应归类为第

$$\arg\min_{o}(d(\boldsymbol{y},\boldsymbol{c}_o)) \tag{5-9}$$

类。其中，$o=1,2,\cdots,k$。

### 5.5.2 线性分类

如前文所述，分类平面可视作训练数据的编码。对线性二分类平面来说，参数向量 $\boldsymbol{w}$ 包含训练数据的编码信息。给定待预测样本 $\boldsymbol{x}$，当 $\boldsymbol{w}[\boldsymbol{x},1]^{\mathrm{T}}>0$ 时，样本 $\boldsymbol{x}$ 被归类为正例；当 $\boldsymbol{w}[\boldsymbol{x},1]^{\mathrm{T}}<0$ 时，样本 $\boldsymbol{x}$ 被归类为负例。显然，预测过程即是对训练数据编码的参数向量 $\boldsymbol{w}$ 的解码过程。

### 5.5.3 纠错输出码

仍以纠错输出码为例，与分类问题拆分与编码相对应类似，各分类器预测结果的集成与解码相对应。如图 5-2 所示，对于预测样本 $\boldsymbol{x}$ 来说，不同的分类器预测其为正负例的情况组成行向量。计算此向量与纠错输入码各编码行的距离，如图 5-2 所示，二者的曼哈顿距离构成图示中的列向量。其中，最小值所在行所属类别即为样本 $\boldsymbol{x}$ 的预测结果。

| | $f_1$ | $f_2$ | $f_3$ | $f_4$ | $f_5$ | |
|---|---|---|---|---|---|---|
| $C_1$ | +1 | −1 | +1 | −1 | −1 | 6 |
| $C_2$ | −1 | +1 | −1 | −1 | −1 | **4** |
| $C_3$ | −1 | −1 | +1 | −1 | +1 | **2** |
| $C_4$ | −1 | +1 | −1 | +1 | +1 | 4 |
| $x$ | −1 | +1 | +1 | −1 | +1 | |

(a) 多对多

| | $f_1$ | $f_2$ | $f_3$ | $f_4$ | |
|---|---|---|---|---|---|
| $C_1$ | +1 | −1 | −1 | −1 | 4 |
| $C_2$ | −1 | +1 | −1 | −1 | 4 |
| $C_3$ | −1 | −1 | +1 | −1 | 4 |
| $C_4$ | −1 | −1 | −1 | +1 | **2** |
| $x$ | −1 | −1 | −1 | +1 | |

(b) 一对其余

| | $f_1$ | $f_2$ | $f_3$ | $f_4$ | $f_5$ | $f_6$ | |
|---|---|---|---|---|---|---|---|
| $C_1$ | +1 | +1 | +1 | 0 | 0 | 0 | 7 |
| $C_2$ | −1 | 0 | 0 | +1 | +1 | 0 | **3** |
| $C_3$ | 0 | −1 | 0 | −1 | 0 | +1 | 9 |
| $C_4$ | 0 | 0 | −1 | 0 | −1 | −1 | 5 |
| $x$ | −1 | +1 | −1 | +1 | +1 | −1 | |

(c) 一对一

图 5-2 纠错输出码解码示例

### 5.5.4 特征降维

对于 PCA 降维来说，设原中心化样本特征向量长度为 $d$，由样本协方差矩阵 $\boldsymbol{X}\boldsymbol{X}^{\mathrm{T}}$ 前 $d'$ 大的特征值对应的特征向量 $\boldsymbol{w}_1,\boldsymbol{w}_2,\cdots,\boldsymbol{w}_{d'}$ 构成的投影矩阵 $\boldsymbol{W}=[\boldsymbol{w}_1;\boldsymbol{w}_2;\cdots;\boldsymbol{w}_{d'}]$，对新样本特征向量 y 进行降维处理，即是对降维压缩编码进行解码处理，新样本特征向量 y 的降维结果为 $\boldsymbol{W}\boldsymbol{y}^{\mathrm{T}}$，其长度为 $d'$。

## 5.6 自 编 码

前文所述特征选择、降维均建立在特征提取基础上。当前流行的神经网络模型，采用由多个隐层构成的自编码器完成对训练数据显著特征的提取。通常地，自编码器由一个编码器和一个解码器构成。编码过程对应一个函数映射 $f$，负责将样本 $\boldsymbol{x}$ 编码为特征 $\boldsymbol{h}$，即 $\boldsymbol{h}=f(\boldsymbol{x})$。有必要说明的是，也有书籍将此编码结果称作隐藏码。另外，有必要指出的是，根据待解决问题的不同，解码器 $\boldsymbol{r}=g(\boldsymbol{h})$ 的形式各不相同。

### 5.6.1 恒等变换

如果解码器对应的函数映射是编码器函数映射的逆映射，即 $g=f^{-1}$，则 $g(f(\boldsymbol{x}))=\boldsymbol{x}$。