摘要

在生物学研究中，DNA 超敏位点(DHSs)能够提供给生物学家大量信息来解读 DNA 序列。近年来，在与计算机、数学之间的学科交叉中，生物学科的进展突飞猛进，已经形成了不少成熟的算法来预测 DNA 超敏位点。

本文即主要研究 DNA 超敏位点预测中的信息提取与预测方法研究，阐述生物学研究中近代的发展以及阻碍，引进一套被研究者们广泛认可遵循的研究流程，建立一组标准的含 DNA 超敏位点和不含 DNA 超敏位点的数据集，再对其以基于二核苷酸的空间自相关方法(DSA)进行特征提取，并用主成分分析法（PCA）对其进行数据降维，提高数据可操作性，接着利用 SMOTE 算法来处理不平衡的正负样本，介绍具体信息提取方法和分类方法原理和步骤，使用支持向量机、朴素贝叶斯和随机森林进行分类训练。

利用机器学习工具，结合具体的代码实现效果，利用十折交叉验证，比较预测方法的交叉组合效果，优化最终表现，最终使用三种不同分类器分别得到了 92.0%，80.3%，85.9%的较为良好的准确率，同时可以求出其混淆矩阵，计算分类评估指标。多次实验之后我们发现，SMOTE 扩充样本后可以很好的改善分类器的精确度。同时支持向量机对分类的准确度最高，适应性好。归纳出在在研究 DNA 超敏位点可用于推广复制的经验，提供部分改进的理论算法，并以期可在三核苷酸和其他研究主题中得到更加广泛的应用。

关键词：DNA 超敏位点 特征提取 SMOTE 算法 随机森林 交叉验证

ABSTRACT

In biological research, DNA hypersensitive sites (DHSs) can provide biologists with a lot of information to interpret DNA sequences. In recent years, through the interdiscipline between biology, computer and mathematics, the progress of the biology department has advanced by leaps and bounds. And many mature algorithms have been formed to predict DNA hypersensitivity sites.

This paper mainly studies information extraction and prediction methods in DNA hypersensitive site prediction, explains the development and obstacles of modern research in biological research, introduces a set of research processes that are widely recognized and followed by researchers. We try to set up a standard data set with those containing DNA hypersensitivity sites and those without DNA hypersensitivity sites. Feature extraction is based on the dinucleotide-based spatial autocorrelation method (DSA), with the Principal Component Analysis (PCA) data to reduce dimension and improve data operability. Then I use SMOTE algorithm to process unbalanced positive and negative samples, introduce principles and steps of specific information extraction methods and classification methods, using support vector machines, naive Bayes and random forest for classification training.

Use machine learning tools, combined with specific code to obtain the consequence, with 10-fold cross-validation, comparing the cross-combination effect of the prediction method, and optimizing the final performance, the final use of three different classifiers are 92.0%, 80.3%, and 85.9% respectively. With good accuracy, at the same time, we can find the confusion matrix and calculate the classification evaluation index. Also, after many experiments, we find that SMOTE can improve the accuracy of the classifier after expanding the sample. Moreover, the support vector machine has the highest classification accuracy and good adaptability. Finally, we summarize the experience that can be used to promote replication in the study of DNA hypersensitivity sites, provide some improved theoretical algorithms, and hope to be more widely used in trinucleotide and other research topics.

Key words：DNA hypersensitive sites Feature extraction SMOTE Random Forest Cross-Validation

目录

一、绪论

1.1 研究背景及意义

在人类的基因组中，含有活性顺式作用元件的 DNA 序列会经历核小体的动态位移，这类元件能够调节蛋白质使得 DNA 序列对 DNase I 酶的裂解变得高度敏感。这些特定的基因组区域被称为 DNase I 超敏感位点(DHSs)，这些位点在 20 世纪 80 年代被识别。DHSs 是含有转录调节因子的染色质区域的可靠和通用标记，对于发现基因调控中涉及的非编码功能性因子和理解基因表达的调控机制至关重要，并且对各种各样的基因组调控元件的发现有直接的推动作用。因此，检测并提供 DHSs 的信息是一种检测功能性调节元件位置的极其准确的方法，给发现大多数实验建立的远端顺式作用元件的打下了基础。印迹杂交是一种非常传统的检测 DHSs 的技术，然而，从这项技术中提取信息不仅价格高昂极费时间而且并不准确。在最近的研究中，一种基于高通量测序的技术已经被广泛的应用于人类基因。遗憾的是，执行全基因组测序需要巨大的花费和劳动力，因此，开发自动化的计算方法并且能高效准确的识别 DHS 刻不容缓。事实上，目前已经有几种计算方法被应用于解决有关解决测序信息的问题。

其中，通过计算机手段得到测序的信息是有极大优势的。对于合理的药物设计而言，蛋白质以及他们的复合物和配体的三维结构知识至关重要。尽管利用 X 射线结晶化是一种确定这些结构强有力的工具，然而其不仅昂贵耗时，并且并不是所有的蛋白质都可以成功的结晶。例如，膜蛋白是难以洁净的，它们中的大多数都难以溶解在普通的溶剂中。因此，迄今确定的膜蛋白结构仍然很少。核磁共振技术(NMR)也是一种非常有效的技术，然而确定

膜蛋白的 3D 结构时它同样成本高昂且耗时耗力。为了及时的获取结构信息，近年来研究者已经开发了相当多的生物结构信息工具。同时，面对在后基因组时代发现的爆炸性增长的生物序列，如何及时的使用他们应用到药物研发中，很多基于序列的重要信息在不同的生物测序工具中被剔除。事实上，生物测序学和生物结构学的快速发展导致药用化学产生了前所未有的革命，其中计算生物学扮演了推动寻找新型药物发展的重要作用。因此，计算机算法同样的也使用在识别 DHS 的研究上。

1.2 研究方法、思路与框架

为了研究一个真正有效的生物系统，我们需要遵循周氏五步法则：

(1)选择或建立一个有效的基准数据集用于训练和测试预测器。

(2)将样本替换成一个有效的算式，其能真正反映他们与要预测的目标的内在相关性。

(3)引进或者研究一种强有力的算法执行预测。

(4)正确执行交叉验证测试，以客观地评估预期的预测精度。

(5)建立一个对用户有好的供公众访问的预测器网页。

在这篇文章中，结构上分为我将遵守以上法则主要阐述基本的对 DNA 超敏位点的预测方法，对前四步骤着力进行阐述说明。

二、实验数据准备

2.1 构建实验数据集

要构建一个有泛用性的计算模型，需要有效的基准数据集来有效地训练和测试模型。因此，我们采用了被研究者广泛使用的数据集 S1 和 S2. S1 包含 280 个 DHS 片段和 738 个非 DHS 片段. 数据集 S1 可以被表述为：

$$S_1 = S_1^+ + S_1^- \qquad \text{式(2-1)}$$

S_1^+ 代表 DHS 的样本数据集，S_1^- 代表非 DHS 片段，∪是一个数学符号代表合并。

数据集 S_2 同理可以表示为：

$$S_2 = S_2^+ + S_2^- \qquad \text{式(2-2)}$$

S_2 包含 247 个 DHS 片段和 710 个非 DHS 片段。

2.2 特征提取

DNA 性质包含物理性质和化学性质，附录表 2.2 是二核苷酸的性质矩阵。其中，物理属性和热力学属性包括延展性，结构性，和热力学性质。在这篇论文中，我们用以下公式归一化了性质矩阵中的每一个元素。

$$\frac{X - X_{min}}{X_{max} - X_{min}} \qquad \text{式(2-3)}$$

X 是原本的性质数值，X_{min} 是最小的性质元素，X_{max} 是最大的性质元素。DNA 序列已经被证实了是由四种核苷酸组成的聚合物，它们分别是腺嘌呤(A)，胞嘧啶(C)，鸟嘌呤(G)，胸腺嘧啶(T)，当我们使用二核苷酸来表示 DNA 序列时，便会产生总共有 4×4=16 种组合物。表格 2.2(见附录 A）展示了 15 种二核苷酸的性质，因此二核苷酸序列可以被转化成一个矩阵 $PRO = p(i, j)_{15\times(L-1)}$，L 代表序列的长度，15 则代表第 i 个二核苷酸对性质的总数。

Geary's C 是一种空间自相关的统计方法，它能将全局序列顺序信息纳入我们的模型之中，因此我们可以利用一种名为基于二核苷酸的空间自相关方法(DSA)来提取特征信息。如图所示，假设我们得到了一组 DNA 序列 D：

$$D=N_1N_2N_3N_4...N_L \qquad \text{式(2-4)}$$

L 指这组 DNA 序列的长度，N_i (i=1，2，…，L)指二核苷酸。如图所示，DSA 可以被表示成：

$$G_{s,t}=\frac{\frac{1}{2(L-g-1)}\sum_{i=1}^{L-g-1}(p_{i,s}-p_{i+g,s})(p_{i,t}-p_{i+g,t})}{\frac{1}{(L-1)-1}\sum_{i=1}^{L-1}(p_{i,s}-\overline{p_s})(p_{i,t}-\overline{p_t})} \qquad \text{式(2-5)}$$

s 和 t 是性质的下标。当 s=t 时，算式表示为序列的自相关性，当 s≠t 时，算式表示为序列的互相关性。$p_{i,s}$ 是二核苷酸 D_i 在位置 i 下标为 s 时的数值，$p_{i,t}$ 是二核苷酸 D_i 在位置 i 下标为 t 时的数值，$\overline{p_s}$ 代表性质 s 的数值的平均值，$\overline{p_t}$ 代表性质 t 的数值的平均值。

三、特征降维

3.1 降维的基本思想

我们在研究某些问题时，需要处理带有很多变量的数据，比如研究房价的影响因素，需要考虑的变量有物价水平、就业率、土地价格、城市化水平等。变量和数据很多，但是可能存在噪音和冗余，因为这些变量中有些是相关的，那么就可以从相关的变量中选择一个，或者将几个变量综合为一个变量，作为代表。用少数变量来代表所有的变量，用来解释所要研究的问题，就能从化繁为简，抓住关键，这也就是降维的思想。

3.2 主成分分析法

主成分分析法能将多个关系不明确的复杂变量，转换为少数几个独立不相关的综合变量，从而来比较全面地反映整个数据集。这是因为数据集中的原始变量之间存在一定的相关关系，可用较少的综合变量来综合各原始变量之间的信息。

3.2.1 算法基本原理

我们先从最简单的情形开始，假定数据集中的原始变量只有两个，即数据是二维的，每个样本数据都用标准的 X-Y 坐标轴来表示。如果每一个维度都服从正态分布，那么这些数据在坐标轴上就会呈现椭圆形的分布，且有正交的长轴和短轴。

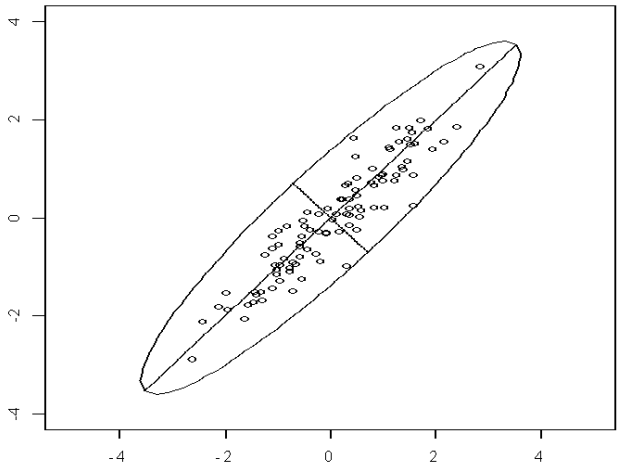

图 3.1 二维数据椭圆点阵

在短轴上，样本数据的变化比较小，如果把这些点垂直地投影到短轴上，那么有很多点的投影会重合，这相当于很多数据点的信息都没有被充分利用到；而在长

轴上，样本的数据变化范围更广。因此，如果我们想把二维数据降维变成一维数据，我们只要把长轴和短轴摆放到坐标轴上，并将短轴退化即可。

因此，我们只需要将椭圆点阵“摆正”到坐标轴上即可。利用坐标变换，把观测点在原坐标轴的坐标转换到新坐标系下，同时也把原始变量转换为了长轴的变量和短轴的变量，这种转换是通过对原始变量进行线性组合的方式而完成的。

比如一个观测点在原 X-Y 坐标系中的坐标为(6，7)，坐标基为(1，0)和(0，1)，如果长轴为斜率是 1 的线，短轴为斜率是 –1的线，新坐标系以椭圆点阵相互正交的长短轴作为 X-Y 坐标轴，那么新坐标基可以取为$(\frac{1}{\sqrt{2}},\frac{1}{\sqrt{2}})$和$\left(-\frac{1}{\sqrt{2}},\frac{1}{\sqrt{2}}\right)$。将坐标基按行摆放作为变换矩阵 P，乘以原坐标，就能求出在此新坐标系下变换出来的新的坐标。可以看到变换后长轴变量的值远大于短轴变量的值。

$$\begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ -1/\sqrt{2} & 1/\sqrt{2} \end{bmatrix}\begin{bmatrix} 6 \\ 7 \end{bmatrix}=\begin{bmatrix} 13\sqrt{2}/2 \\ 1/\sqrt{2} \end{bmatrix} \qquad \text{式(3.1)}$$

如果长轴变量解释了数据集中的大部分变化，那么就可以用长轴变量来代表原来的两个变量，从而把二维数据降至一维。椭圆的长轴和短轴的长度相差越大，这种做法的效果也就越好。

将二维变量推广到多维变量，具有多维变量的数据集其观测点的形状类似于一个高维椭球，同样的，把高维椭球的轴都找出来，再把代表数据大部分信息的 k 个最长的相互垂直的轴作为新变量，也就是 k 个主成分，这便初步找到了主成分。选择的主成分越少，越能体现降维二字的内涵，可是不可避免会舍弃越多的信息。因此需要制定一定的标准来决定我们应该选择几个，选择哪几个主成分。

假定我们有 m 个样本数据，每个样本数据有 n 个特征，将其排成 n 行 m 列的矩阵，将其进行标准化，求出矩阵 X。用 X_1， X_2，…， X_n 来表示 n 个原始变量：

$$X=\begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1m} \\ x_{21} & x_{22} & \cdots & x_{2m} \\ \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ x_{n1} & x_{n2} & \cdots & x_{nm} \end{bmatrix}=\begin{bmatrix} X_1 \\ X_2 \\ \cdots \\ \cdots \\ \cdots \\ X_n \end{bmatrix} \qquad \text{式(3.2)}$$

由上式我们可以求出 n 个新变量，其手段是利用变换矩阵 P 对原数据向量进行矩阵变换。转换矩阵可以有很多个，变换的坐标系有很多个，但是只有一个可以由原始变量得到主成分。假定我们已经得到了转换矩阵 P，把转换后的 n 个主成分记为 Y_1，Y_2，…，Y_n，由 Y=PX，就可以得到主成分矩阵：

$$Y=[Y_1,Y_2,Y_3,...,Y_n]^T \qquad \text{式(3.3)}$$

这 n 个行向量都是主成分，彼此之间是线性无关的，按照对数据变化解释力的强度降序排列，然而并非被挑出来的前 k 个行向量才叫做主成分。

当 n=2 时，主成分为 Y_1 和 Y_2，原变量为 X_1 和 X_2。从下图可见 Y_1 为长轴变量，数据沿着这条轴的分布比较分散，数据的变化比较大，因此可以用 Y_1 作为第一主成分来替代 X_1 和 X_2。

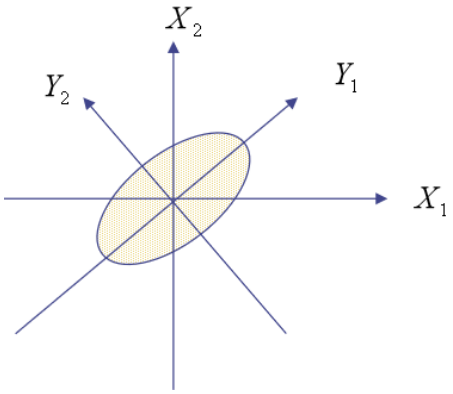

图 3.2 主成分与原变量坐标关系

我们用方差来量化数据的变化和分散程度。把向量 X_1 和 X_2 的元素记为 x_{1a}，x_{2a}(a=1，2，...，m)，把主成分 Y_1 和 Y_2 的元素记为 y_{1a}、y_{2a}(a=1，2，...，m)，那么整个数据集上的方差可以如下表示：

$$\begin{aligned}\frac{1}{m}\sum_{a=1}^{m}(x_{1a})^2+\frac{1}{m}\sum_{a=1}^{m}(x_{2a})^2&=\frac{1}{m}\sum_{a=1}^{m}(y_{1t})^2+\frac{1}{m}\sum_{a=1}^{m}(y_{2a})^2\\&=\alpha\cdot Var+(1-\alpha)\bullet Var\end{aligned} \qquad \text{式(3-4)}$$

由于之前的标准化的操作，所以行向量的均值为 0。

第一主成分 Y_1 所能解释的数据的变化，可以用主成分的方差来衡量：

$$\sigma^2=\frac{1}{m}\sum_{a=1}^{m}y_{1a}^2 \qquad \text{式(3-5)}$$

也可以用主成分的方差占总体方差的比重来衡量，这里假设为$\alpha,\alpha\in(0,1)$。其值越大，则反映的信息越多。在有 n 个原始变量和 n 个主成分的例子中，选择合适的转换矩阵 P 来计算得到主成分矩阵 Y 时，要让单个主成分在数据集上的方差尽可能大。选择主成分的一般指标是少数 k 个主成分(1≤k<n)的方差，其方差之和与总体数据的方差的比例不超过α，α一般取值为 85%-95%。

如果已知转换矩阵 P 和主成分矩阵 Y，利用上述的主成分的方差占比来衡量该主成分能解释的数据集方差的大小，然后按这个比例从大到小进行排序，并进行累加，如果到第 k 个主成分时，累加的比例恰好等于或者超过α，那么就选择这 k 个主成分作为新变量，对数据集进行降维。因此最后我们只需要结局一个问题，求解转换矩阵 P。

主成分矩阵 Y 的一个特点是，单个主成分向量 Yi 的方差占总体方差的比例尽可能大，同时主成分由于上述的原理可以降序排列。另外，还有一个特性是所有的主成分都是线性无关的，或者说是正交的，那么所有主成分中，任意两个主成分 Y_i 和 Y_j 的协方差都是 0。

第一个特点涉及到主成分的方差，第二个特点涉及到主成分之间的协方差，这自然而然让我们想到协方差矩阵。因主成分矩阵 Y 的协方差矩阵的对角元素是每个主成分的方差，而其余的元素分别是两两特征的协方差。当协方差为 0 时，协方差矩阵除了对角元素其余元素均为 0，便意味着这是一个对角矩阵。

由于数据集已经经过标准化，那么同样，主成分矩阵中的行向量也是 0 均值的，于是某两个主成分的协方差为：

$$d_{ij}=\frac{1}{m}\sum_{a=1}^{m}(y_{ia}y_{ja})=\frac{1}{m}Y_iY_j^T \qquad \text{式(3-6)}$$

进一步得到主成分矩阵 Y 的协方差矩阵为：

$$D=\frac{1}{m}\begin{bmatrix}Y_1Y_1^T & \dots & Y_1Y_n^T\\ \dots & Y_iY_j^T & \dots\\ Y_nY_1^T & \dots & Y_nY_n^T\end{bmatrix}=\begin{bmatrix}d_1 & \dots & 0\\ \dots & d_j & \dots\\ 0 & \dots & d_n\end{bmatrix}=\frac{1}{m}YY^T \qquad \text{式(3-7)}$$

知道了主成分矩阵 Y 的协方差矩阵在经过矩阵变换后实质上是一个对角矩阵，把 Y=PX 这个等式代入协方差矩阵中进行变换，已知的数据 X 和需要求的 P 都放到了协方差矩阵中：

$$D=\frac{1}{m}YY^T=\frac{1}{m}(PX)(PX)^T=\frac{1}{m}(PXX^TP^T)=P(\frac{1}{m}XX^T)P^T \qquad 式(3\text{-}8)$$

通过原数据集 X 的协方差矩阵 $C=\frac{1}{m}XX^T$ 我们可以得到主成分矩阵 Y 的协方差矩阵。

数据集 X 的协方差矩阵 $C=\frac{1}{m}XX^T$ 作为 n 阶实对称矩阵，根据其性质一定可以找到 n 个单位正交特征向量将其相似对角化。设这 n 个单位正交特征向量分别为 e_1，e_2，…，e_n，并将其排列成一个矩阵：

$$E=(e_1,e_2,...,e_n) \qquad 式(3\text{-}9)$$

数据集 X 的协方差矩阵则可以对角化为：

$$E^T(\frac{1}{m}XX^T)E=\Lambda=\begin{bmatrix}\lambda_1 & & & \\ & \lambda_2 & & \\ & & ... & \\ & & & \lambda_n\end{bmatrix} \qquad 式(3\text{-}10)$$

相似对角阵上的元素 $\lambda_1,\lambda_2,...,\lambda_n$ 是协方差矩阵的特征值，其对应的单位特征向量是 E 中的列向量。我们把这个对角阵 Λ 上的元素按照值的大小从高到低排列。同时把和单位特征向量对应的矩阵 E 里的列向量也进行排列。我们假设上面已经是排列好之后的形式，由于主成分矩阵的协方差矩阵也是元素从大到小降序排列的对角矩阵，就可以得到：

$$D=\frac{1}{m}YY^T=\begin{bmatrix}\lambda_1 & & & \\ & \lambda_2 & & \\ & & ... & \\ & & & \lambda_n\end{bmatrix}=P(\frac{1}{m}XX^T)P^T=E^T(\frac{1}{m}XX^T)E \qquad 式(3\text{-}11)$$

也就是取 X 的协方差矩阵的单位特征向量矩阵 E，用它的转置 E^T 来作为转换矩阵 P，而 X 的协方差矩阵的特征值 λ 就是各主成分的方差。由 Y=PX 我们自然

就可以得到主成分矩阵 Y。如果我们想把数据从 n 维降至 k 维，那么从 P 中挑出前 k 个行向量，去乘以数据集 X 就行，就可以得到前 k 个主成分。

重新审视主成分的组成成分。可以看出是由协方差矩阵的单位特征向量和原始变量进行线性组合得到的。

$$\begin{aligned} Y &= PX \\ &= \begin{bmatrix} P_1 \\ P_2 \\ \cdots \\ P_n \end{bmatrix} \cdot \begin{bmatrix} X_1 \\ X_2 \\ \cdots \\ X_n \end{bmatrix} \\ &= \begin{bmatrix} e_{11} & e_{12} & \cdots & e_{1n} \\ e_{21} & e_{22} & \cdots & e_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ e_{n1} & e_{n2} & \cdots & e_{nn} \end{bmatrix} \cdot \begin{bmatrix} X_1 \\ X_2 \\ \cdots \\ X_n \end{bmatrix} \\ &= \begin{bmatrix} e_{11}X_1 + e_{12}X_2 + \ldots + e_{1n}X_n \\ e_{21}X_1 + e_{22}X_2 + \ldots + e_{2n}X_n \\ \cdots \\ e_{n1}X_1 + e_{n2}X_2 + \ldots + e_{nn}X_n \end{bmatrix} \end{aligned} \qquad \text{式(3-12)}$$

以上我们可以看出，第一主成分就是：

$$Y_1 = e_{11}X_1 + e_{12}X_2 + \ldots + e_{1n}X_n \qquad \text{式(3-13)}$$

且第一主成分的方差是最大的。

然后第二主成分满足：(1)和第一主成分正交，(2)在剩余的其他主成分中，方差最大，表达式为：

$$Y_2 = e_{21}X_1 + e_{22}X_2 + \ldots + e_{2n}X_n \qquad \text{式(3-14)}$$

同理，第 k 个主成分我们同样也可以这样求解出：

$$Y_k = e_{k1}X_1 + e_{k2}X_2 + \ldots + e_{kn}X_n \qquad \text{式(3-15)}$$

我们知道用主成分的方差来衡量其所能解释的数据集的方差，而主成分的方差就是 X 的协方差矩阵的特征值 λ，所以第 k 个主成分的方差就是 λ_k。它是第 k 个主成分的方差占总方差的比例：

$$\frac{\lambda_k}{\lambda_1+\lambda_2+...+\lambda_n} \qquad \text{式(3-16)}$$

我们可以将其称作：主成分 Y_k 的方差贡献率。

那么前 k 个主成分的方差贡献率之和我们同样可以表达为：

$$\frac{\lambda_1+\lambda_2+...+\lambda_k}{\lambda_1+\lambda_2+...+\lambda_n} \qquad \text{式(3-17)}$$

如果选择的前 k 个主成分的方差之和贡献率超过了之前界定的值α，那么说明用前 k 个主成分去代替原来的 n 个变量后，不能解释的方差不足$1-\alpha$，没有损失太多信息，于是我们可以把 n 个变量减少为 k 个变量，达到降维的目的。

3.2.2 算法流程

综上所述，假设我们拿到了一份数据集，有 m 个样本，每个样本由 n 个特征来描述，那么我们可以按照以下的步骤进行降维：

1、将数据集中的每个样本作为列向量，按列排列构成一个 n 行 m 列的矩阵；

2、将矩阵的每一个行向量进行标准化，即用原来的值减去此行向量的均值，得到新的数据集矩阵 X；

3、求 X 的协方差矩阵 $C=\frac{1}{m}XX^T$。求出协方差矩阵的特征值 λ。计算对应的单位特征向量 e；

4、将上述的单位特征向量以对应的特征值从高到低的顺序重新排列成矩阵，得到转换矩阵 P，并按 PX 计算出主成分矩阵；

5、计算方差累计贡献率，取方差累计贡献率超过α的前 k 个主成分，或者想降至特定的 k 维，直接取前 k 个主成分。

3.3 Lasso 回归降维

我们先作一个基本的假设：稀疏性假设。简单来说，我们认为，尽管世界如此复杂，但有用的信息却非常有限。套用到常见的统计学模型中，假如，我们考虑一个线性回归模型，有一个因变量 Y，但有成百上千的自变量 X。我们假定，只有有

限个 X 的回归系数不为 0，但其余的都是 0。也就是说他们跟 Y 并没有特别显著的关系。找到其中重要的 X，对我们理解数据有重要的意义。

对应前文的例子，生物学家想要研究基因对于某类疾病的影响，面对上万个可能的基因，生物学家们倾向于认为只有其中的一小部分对于该类疾病有着显著的影响；而为了预测消费者对于电影和书籍的喜好，线上电影和书店也倾向于认为一个消费者的喜好可以从少量的评分数据中得到。而“稀疏性假设”就是对于人们这种倾向的具体体现，举个例子，在分析基于和疾病的关系时，对于我们放入的 10000 个可能的基因，我们认为这 10000 个基因在回归模型中的回归系数只有少量的不为 0。在高维变量和稀疏性假设的情况下，Lasso 算法应运而生。

3.3.1 算法基本原理

Lasso 相比于普通最小二乘估计，可以在变量众多的时候快速有效地提取出重要变量，简化模型。这些变量有的是“关键变量”，有的无显性关系，与变化关系不大。对应到回归问题中：

$$Y=\beta_0+\beta_1X_1+\beta_2X_2+\beta_3X_3+...+\varepsilon \qquad \text{式(3-18)}$$

Lasso 最终可以使得部分对应的自变量 X 的系数取值为 0，代表相应的自变量跟 Y 关系不大。

Lasso 虽然似乎是个完美的算法，可以产生稀疏估计，舍弃无关变量。但是，实际上产生的是有偏估计。实际上通过 Lasso 算法我们可以发现：

1、虽然最小二乘估计是无偏估计，但是在变量过多的情况下，往往带有较大的方差。我们考虑分解熟悉的均方误差(MSE)：

$$MSE=\sigma^2+b^2 \qquad \text{式(3-19)}$$

也就是 MSE 由偏差 bias 和方差 σ^2 两部分组成；这两部分很难同时缩小。Lasso 虽然是有偏估计，但是在引入一定的偏差的同时，可能可以大量降低估计的方差，从而降低整体的均方误差。其优点不言而喻：如果我们拥有的样本信息是有限的，那么我们想要用有限的信息去估计过多的系数，此时信息很可能会出现不够用的情况，所以筛选变量提高估计效果十分必要。

2、对于最后得到的回归方程，我们需要在估计出每一个放入模型的自变量的系数后，能够更好地解释它。当得到稀疏估计之后，我们就能够明确得知：到底哪些关键变量对结果有显著的影响，从而更好地提高降维质量。

通过 Lasso 的表达式：

$$\beta = \arg\min_{\beta} \sum_{i=1}^{N} (y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 \quad \text{s.t.} \sum_{j=1}^{p} |\beta_j| \le t \qquad \text{式(3-20)}$$

或者：

$$\beta = \arg\min_{\beta} \left\{ \frac{1}{2} \sum_{i=1}^{N} (y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \right\} \qquad \text{式(3-21)}$$

两个式子是等价的，但在实际应用中更多地提到第二个式子。在第一个式子中，第一行是我们熟知的最小二乘法的目标函数，但在 Lasso 的估计过程中加上了第二行的限制条件，这个限制条件就对应与第二个式子中的第二项。t 越小，或者 λ 越大，对估计参数的压缩作用就越强。当我们对这个目标函数求最小时，限制效果不显著的自变量便被降维，其系数取值变成了 0。

3.3.2 算法对比

我们以二维的情况为例，如下图左所示，图中椭圆表示上式第一项 β 不同的取值时的图像，越靠近椭圆中心取值效果越好，同一个椭圆上的值相同。而图中以原点为中心的正方形则表示满足一式第二行的限制条件的点集，所以我们也只能选取落入该正方形的点。最终 Lasso 的估计值为椭圆和下面矩形的交点，除非椭圆与矩形正好相切在矩形的某条边上，否则交点将落在矩形的顶点上，这时某参数的估计值将被压缩到 0，即该变量已被剔除出模型。

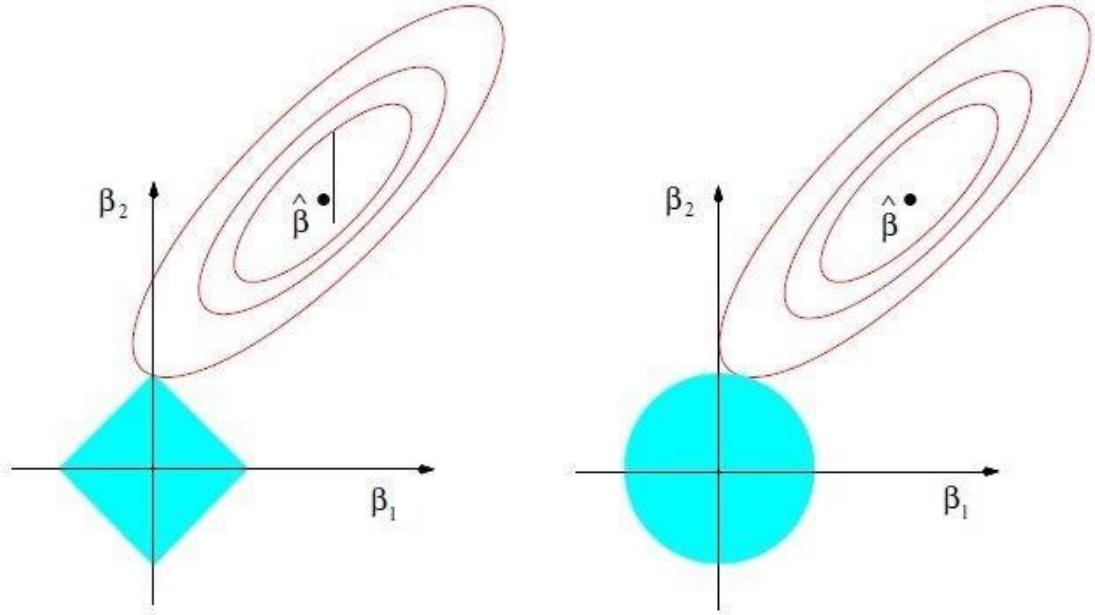

图 3.3 Lasso 回归(作)与岭回归(右)

而图右则是经常来来和 Lasso 进行对比的岭回归，其计算方法如下：

$$\beta^{ridge} = \arg\min_{\beta} \sum_{i=1}^{N} (y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2, \quad s.t. \sum_{j=1}^{p} \beta_j^2 \le t \qquad \text{式(3-22)}$$

其与 Lasso 的不同在于第二行的限制条件，Lasso 限制各系数绝对值之和，我们称其为 L1 惩罚函数，而岭回归则是限制各系数的平方和，对应的我们称其为 L2 惩罚函数，所以在二维的情况下，其可行域为以原点为圆心的圆，其最终的估计值也为不断扩大的椭圆与可行域的第一个交点。我们可以发现，由于可行域从矩形变成的圆，其交点将为椭圆与圆的切点，且难以刚好落在坐标轴上。这也使得岭回归很多时候并不能将多余变量的系数压缩为 0。

除了 Lasso 对应的 L1 惩罚以及岭回归对应的 L2 惩罚，还有许多著名的惩罚函数。比如，在牺牲了惩罚函数凸性的情况下，一些非凸的惩罚函数(如 SCAD、MCP 等)能够获得渐近无偏的估计，与之而来的是较高的计算成本。

通过下面这幅图我们能了解到 Lasso 筛选变量的动态过程，以及更直观地了解“压缩估计参数”的意思：

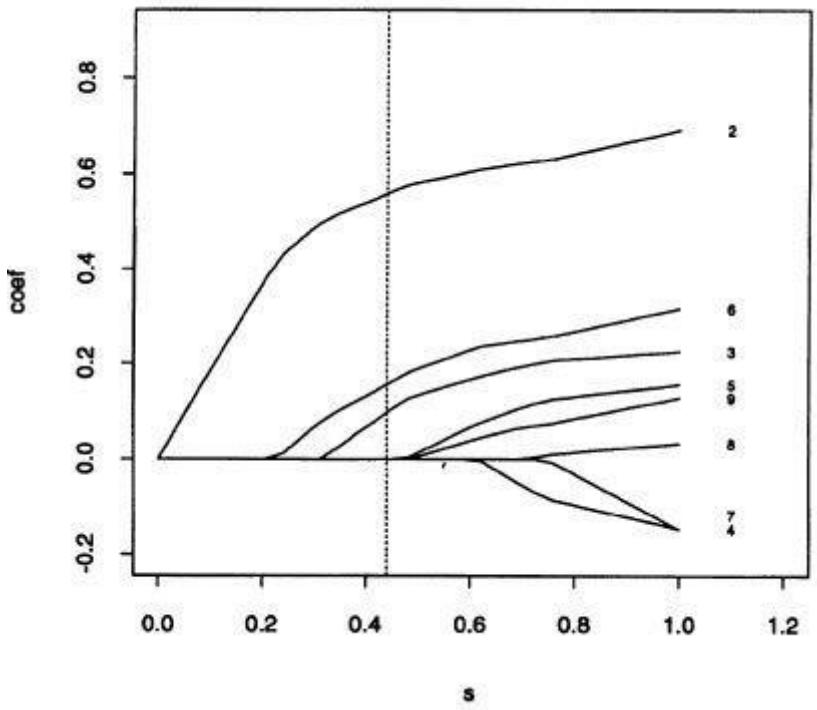

图 3.5 Lasso 筛选变量动态过程图

s 为 t 除以最小二乘法估计值的绝对值之和。t 越小表示限制越强，即 s 越小，图中垂直于横轴的虚线也更靠左，而图中其余曲线则代表在指定 s 的条件下，该变量系数的估计值。当我们确定 s 值后，各估计参数的值即为 x=s 的直线与图中其他曲线交点的横坐标。我们可以看到，当 s 越小，各估计参数相应的也被压缩得更小，而当 s 达到一定值以后，一部分不重要的变量将被压缩为 0，代表该变量已被剔除出模型，图中从右至左不断下降的曲线如同被不断减小的 s 一步一步压缩，直到压缩为 0。在实际中，往往不会使用上面的 s，而是选择用 λ 作为压缩参数。越大的 λ 一般对应越严格的限制。如果选不好 λ，模型的表现会变得非常糟糕了。

3.3.3 参数选取

现在有许多研究者都在研究如何选取 λ，常见使用的准则有交叉验证，会在下文中阐述原理，另外还有 AIC 准则， BIC 准则。当存在真模型的假设下，常常选取 BIC 准则。

AIC 准则指最小化信息量准则，它是对于模型拟合过程中精确度提出的一种标准。通常情况下，它被定义为：

$$AIC = 2k - 2\ln(L) \qquad \text{式(3-23)}$$

其中 k 代表模型中的参数数量，L 代表相应的似然函数。

AIC 能够为参数选取提供了有效的指导，但同样，也存在一定的限制。倘若样本的数量过大，在 AIC 准则中拟合误差会因为样本大小而受到影响，样本越大，则误差会越大。而选取的惩罚因子却不受影响。所以倘若当样本数目膨胀， AIC 准则往往效果不尽如人意，未知的参数可能会比真实模型所含的数量多。

BIC 贝叶斯信息准则是一项根据贝叶斯理论提出的模型准则。这项准则弥补了 AIC 的不足。BIC 的定义为：

$$BIC = k\ln(n) - 2\ln(L) \qquad \text{式(3-24)}$$

其中，k 为模型参数个数，n 为样本数量，L 为似然函数。

AIC 和 BIC 的公式中前半部分趋同，后半部分也都是惩罚项。但不同的是 BIC 相比 AIC，在数据膨胀时对模型参数惩罚力度更大。所以 BIC 准则在数据数目极大时更愿意选择参数少的模型。

相比于如今 Lasso 的“家喻户晓”，在其发明的头几年人气却不是很旺。其大致原因有三点：1、缺少能快速求解 Lasso 的算法；2、新鲜出炉的 Lasso 还未能被大家所理解；3、当时对于高维数据的研究还没有火起来。生物技术的发展带来了高维问题分析场景。关于 Lasso 问题是如何高效运算输出结果。

在最初的论文实现中，Lasso 计算效率比较低，是算法应用的很大瓶颈。这个问题后来被研究者提出的最小角回归算法(LARS)解决，其吸纳了前向梯度算法的优势，即在精确度上有一定的保证，又通过角平分线快速步进，大大提高了计算效率，并且给出了优秀的几何性质的解读，随后被应用于全基因组关联研究[1]。

[1] Sun， J.， Wu， Q.， Shen， D. *et al.* TSLRF： Two-Stage Algorithm Based on Least Angle Regression and Random Forest in genome-wide association studies. *Sci Rep* **9**， 18034 (2019)

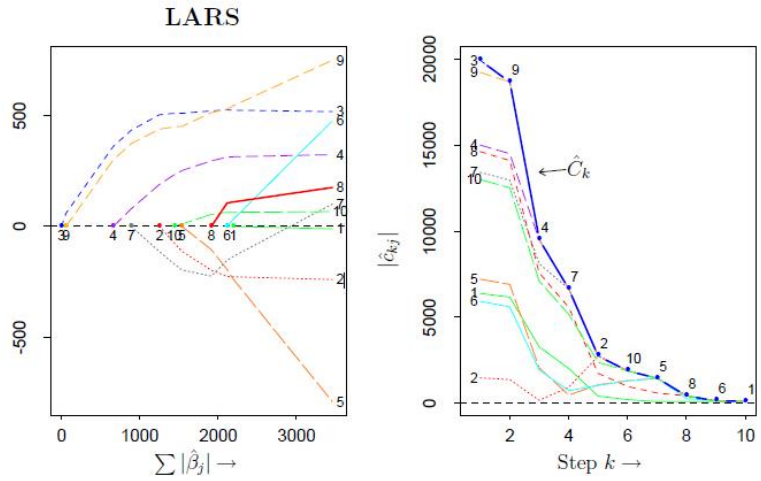

图 3.6 LARS 步进计算结果

四、不平衡数据处理

4.1 出现原因及背景

我们的数据集 S_1 和 S_2 由于正样本数量明显少于负样本的数量，因此是不平衡的。不平衡的数据在分类时会导致偏差。因此，普遍采用多种方法例如 KPCA，FUS 之类的算法对数据进行再处理，其中，SMOTE 算法已经被广泛应用于生物信息预测系统之中。

4.2 SMOTE 算法

4.2.1 算法步骤

它是一种根据过拟合算法改进的算法，其基本思想是分析少量样本并添加新的样本数据进入数据集。一般，SMOTE 算法遵循以下三个步骤：

(1) 对每一个少数类中的样本 x，先求出它到少数类样本集中不同样本的距离，求出其 k 个近邻。出于对每个特征进行同样的考虑，我们采用标准欧氏距离来样本之间的距离。假设在 n 维欧氏空间中有两个样本 $\vec{p}(p_1,p_2,p_3,...,p_n)$ 和 $\vec{q}(q_1,q_2,q_3,...,q_n)$，那么我们可以求出两者之间的欧氏距离(ED)：

$$ED(\vec{p},\vec{q})=\sqrt{\sum_{k=1}^{n}(p_k-q_k)^2} \qquad 式(4\text{-}1)$$

(2) 按照原样本正负样本数量的比例设置一个采样的比例，从少数类样本 x 其 k 近邻中任意地选择若干个不同的样本。对于每一个随机选出的近邻 x_n，分别与原样本通过以下的算式形成新的样本。

$$x_{new}=x+rand(0,1)\times(x-x) \qquad 式(4\text{-}2)$$

(3) 重复步骤(2)N 次以产生 N 个新的样本。

在使用 SMOTE 算法后可以令正样本的数量与负样本的数量接近一致。

4.2.2 算法弊端

SMOTE 在实际处理的时候，容易产生泛化过度的现象，究其原因，其合成样本的过程存在一定的缺陷。简单来说，其虽然对于每个原本的少数类样本都能产生

合成数据样本，但是并不考虑其邻近样本是怎样分布的，这就会增大类间发生重复的风险。

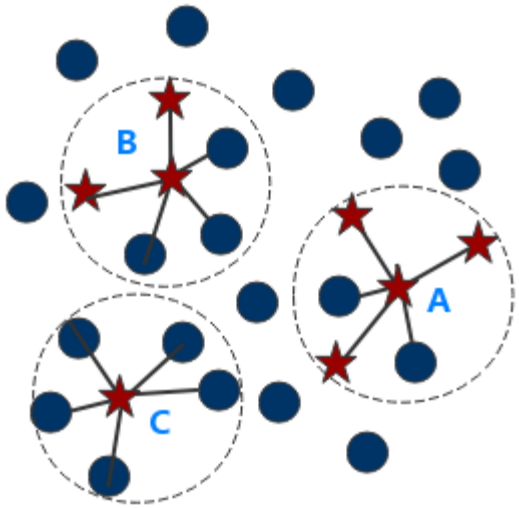

图 4.1 SMOTE 合成样本不同情况

结合上图发现，SMOTE 算法在产生新的少数类样本过程时，插值仅仅发生在同类的近邻之间。 五星 B 和 C 分布在多数类样本周围，但它们合成的少数类样本极有可能被 SMOTE 断定为多数类样本，因为从距离上算法认为它们和多数类的样本更近。因此 SMOTE 算法合成样本较为武断和粗糙。

为了克服这个限制，出现了一些改良的抽样算法。其中现在被运用较多效果较好的有 Borderline-SMOTE 算法。另外，还有对抽样数量本身进行限制自适应合成抽样算法 (ADASYN)等等。

4.2.3 改进的 Borderline-SMOTE 算法

在 Borderline-SMOTE 算法中，想要判断少数类样本，我们需要重新考虑邻样本之间的距离并进行比对：

(1) 首先，对于属于少数类 S_{min} 的样本 x_i 确定一系列最近邻样本集，称该数据集为 $S_{i:m-NN}$，且 $S_{i:m-NN} \subset S$

(2) 接着，对每一个样本 x_i，我们求出最近邻样本集和多数类样本相交的样本的个数：$|S_{i:m-NN} \cap S_{maj}|$

(3) 最后，选择满足下面不等式的 x_i：$\frac{k}{2} \lhd |S_{i:m-NN} \cap S_{maj}| < k$

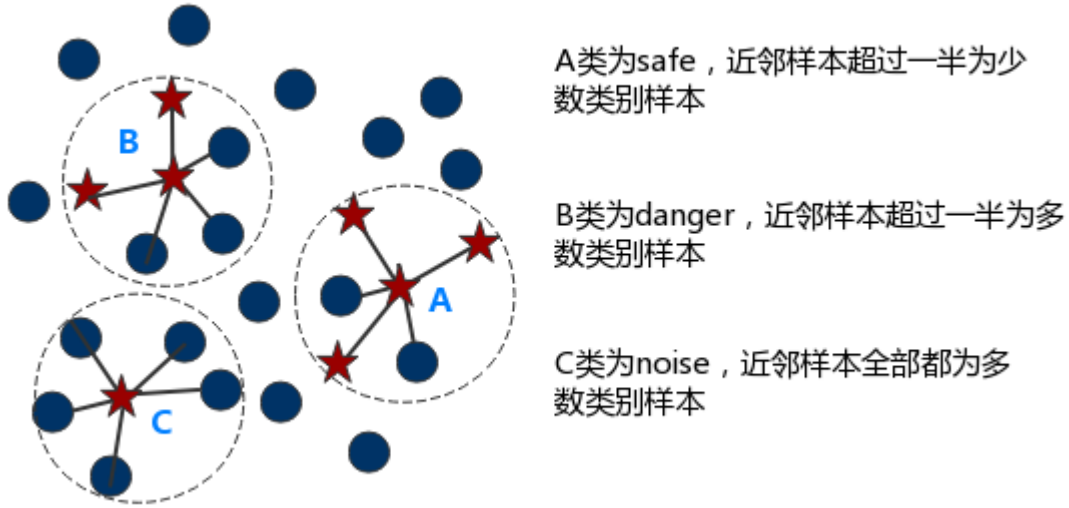

图 4.2 Borderline-SMOTE 对样本进行识别

结合图 4.2 上式表明，当邻集中少数类少于多数类时，符合条件的 x_i 才会被选中形成 “危险集 (DANGER)”。被分类进 DANGER 集中，则意味着此样本处于少数类样本的临界。倘若样本属于 DANGER 集，便对其进行再采样。倘若 x_i 的所有 k 个最近邻样本都属于多类，也就是说 $|S_{i:m-NN} \cap S_{maj}| = k$，那么我们就认为样本 x_i 是属于 NOISE 的，不能生成合成样本。

SMOTE 算法对所有的少数类样本都可以生成合成的样本。Borderline-SMOTE 算法则是有选择的合成样本。这也是 SMOTE 和 Borderline-SMOTE 算法最大的不同。

设训练集为 T，少数类样本 $F = \{f_1, f_2, ..., f_n\}$，Borderline-SMOTE 算法具体描述如下：

步骤一：

1. 求出少数类样本在所有样本中的 k-近邻。

2. 按照 k-近邻对少数类样本重新分类：

该样本为噪声样本(noise),若 k-近邻中绝大多数都是多数类，分类为 N’。

该样本为正常分类样本，若 k-近邻中绝大多数都是少数类，分类为 S。

该样本为临界样本需要再次划分，若 k-近邻中多数类与少数类样本数量接近，分类为 B。

步骤二：

1. 设边界样本集 $B=\{f_1^{'},f_2^{'},...,f_b^{'}\}$，计算 B 集合中的每一个样本 $f_i^{'}$(i=1，2，…，b)在少数类样本中的 k'-近邻 f_{ij}。
2. 随机选出 s(1<s<b)个最近邻。
3. 计算出它们各自与该样本之间的全部属性的差值 $d_{ij}:d_{ij}=f_i^{'}-f_{ij},j=1,2,...,s$
4. 设 $r_{ij},r_{ij}\in(0,1)$ 为一个随机数。用 r_{ij} 与之相乘。倘若 f_{ij} 之前被分类进 N' 或 S 中，则随机数的数值被缩小为 $r_{ij}\in(0,0.5)$
5. 合成少数类样本： $h_{ij}=f_i^{'}+r_{ij}\times d_{ij},j=1,2,...,s$

步骤三

为了均衡样本，不断地进行步骤二，算法结束循环时，已合成需求的少数类样本的数量

4.4 ADASYN 自适应综合过采样法

ADASYN 自适应综合过采样法作为一种插值算法，由数据实际的分布，为少数类样本合成数量不等的新样本。首先计算需要合成的新少数类样本数量，其依据是样本原来的平衡程度，然后再为每个少数类样本计算分布比例。这种算法可能会在多数类样本中间插值出一个少数类样本导致类别重叠。

4.4.1 算法步骤

(1) 记少数类样本为 m_s，多数类为 m_l，计算类不平衡度：

$$d=\frac{m_s}{m_l} \qquad \text{式(4-3)}$$

其中， $d\in(0,1)$

(2) 计算需要合成的样本数量：

$$G=(m_l-m_s)\times\beta \qquad \text{式(4-4)}$$

$\beta \in [0,1]$用于表示在合成样本数据后的期望的平衡水平。当$\beta = 1$时，即 G 等于少数类和多数类的差值，此时合成数据后的多数类个数和少数类数据正好平衡。

(3) 对每个属于少数类的样本$x_i \in S_{min}$在欧氏距离下计算 K 个近邻，Δ为 K 个近邻中属于多数类的样本数目， 记比例r_i：

$$r_i = \frac{\Delta_i}{K}, i = 1,..., m_s \qquad \text{式(4-5)}$$

其中$r_i \in [0,1]$。Δ_i为 K-近邻中样本属于多数类的数量。。

(4) 根据表达式：

$$r_i = \frac{r_i}{\sum_{i=1}^{m_s} r_i} \qquad \text{式(4-6)}$$

对r_i进行标准化，因此r_i满足和为 1 的密度分布($\sum_i r_i = 1$)。计算每个少数类样本的近邻多数类的情况。

(5) 对每个少数类样本计算合成样本的数目：

$$g_i = r_i \times G \qquad \text{式(4-7)}$$

由(2)我们已知 G 是需要合成数据样本的总数，

(6) 在每个带合成的少数类样本周围 K-近邻中任意抽取一个少数类样本，按照：

$$s_i = x_i + (x_{zi} - x_i) \times \lambda \qquad \text{式(4-8)}$$

的表达式生成数据g_i次。$(x_{zi} - x_i)$是一个 n 维空间中的差异向量，λ是一个属于[0，1]的随机数。

五、分类器算法

5.1 随机森林算法

5.1.1 Bootstrap 采样

人们总是希望通过有限的样本数量来推测整体的信息量。Bootstrap 方法将样本当作整体，由样本的统计量来代替整体。这种方法看似存在问题，但是在大量反复的抽取样本后，又是非常合理的。

Bootstrap 步骤：

(1) 在一个数据可以被多次重复抽到的前提下，在原有的样本中抽取某个数量(例如200)的新样本。

(2) 基于产生的新样本，计算出我们需要估计的统计量。

(3) 多次重复步骤(1)(2)n 次，我们就可以得到 n 个统计量的估计值。

(4) 最后，可以利用这不止一个的估计值，估计许多其他的统计量。

5.1.2 Bagging 集成算法

Bagging 即套袋法，用多组训练样本进行多组训练对分类进行投票，从而达到一个“集思广益”的效果。其算法步骤一般遵循以下过程：

（1） 从原始样本集中抽取训练集，利用上述 Bootstrap 算法抽取 n 个训练样本，进行 k 轮 de 抽取，得到 k 个训练集。我们可以认为，这 k 个训练集之间是相互独立的。

（2） 利用多种分类器，可以从 k 个训练集中得到 k 个模型。

（3） 以上 k 个模型采用投票得到分类最终结果；

一方面，由于 $E\left[\frac{\sum X_i}{n}\right]=E[X_i]$，所以 Bagging 算法不能显著降低偏差；另一方面，当子模型相互独立，$Var(\frac{\sum X_i}{n})=\frac{Var(X_i)}{n}$，方差缩小了 n 倍。当子模型之间有一定的关联时，设方差为 σ^2，两两变量的相关性为 ρ $(0<\rho\leqslant 1)$，则：

$$\mathrm{Var}\left(\frac{\sum \mathrm{X_i}}{\mathrm{n}}\right)=\rho\times\sigma^2+(1-\rho)\times\frac{\sigma^2}{\mathrm{n}}\le\sigma^2 \qquad \text{式(5-1)}$$

对方差仍然起到了一定的减小的作用。

5.1.3 CART 决策树

决策树是一种机器学习中典型有标签有监督的分类算法，通过给出一堆样本，每个样本都有一组属性和一个分类结果，也就是分类结果已知，那么通过学习这些样本得到一个决策树，这个决策树能够对新的数据给出正确的分类。决策树的生成主要分一下两步，这两步利用训练样本集来实现：

(1) 分裂节点：当最底层的节点代表的属性无法进行分类时，便会分裂此节点生成 n 个子节点。

(2) 确定阈值：选择合适的阈值控制分类的错误率，降低犯错的概率。

基尼系数是模型不纯度的一种指标，CART 决策树算法采用基尼系数来选择特征从而分类。越低的基尼系数代表越低的不纯度，代表特征选择效果越好。

假设我们有 k 个不同的分类标签，第 k 个分类的概率为$\mathrm{p_k}$，概率分布的基尼系数为：

$$\mathrm{Gini(p)}=\sum_{\mathrm{k=1}}^{\mathrm{K}}\mathrm{p_k}(1-\mathrm{p_k})=1-\sum_{\mathrm{k=1}}^{\mathrm{K}}\mathrm{p_k^2} \qquad \text{式(5-2)}$$

对于样本$\mathbb{Q}$，样本个数为 S，假设第 k 个分类的数量为$\mathrm{C_k}$，其基尼系数为：

$$\mathrm{Gini}(\mathbb{Q})=1-\sum_{\mathrm{k=1}}^{\mathrm{K}}\left(\frac{\mathrm{C_k}}{\mathrm{S}}\right)^2 \qquad \text{式(5-3)}$$

对于样本$\mathbb{Q}$，样本个数为 S，根据特征 T 的某个值 t，把$\mathbb{Q}$分成$\mathrm{Q_1}$和$\mathrm{Q_2}$，则在特征 T 的条件下，样本$\mathbb{Q}$的基尼系数为：

$$\mathrm{Gini}(\mathbb{Q},\mathrm{T})=\frac{\mathrm{Q_1}}{\mathrm{S}}\times\mathrm{Gini}(\mathrm{Q_1})+\frac{\mathrm{Q_2}}{\mathrm{S}}\times\mathrm{Gini}(\mathrm{Q_2}) \qquad \text{式(5-4)}$$

比较基尼系数和熵模型的表达式，二次运算比对数简单很多。尤其是二分类问题，更加简单。对于二类分类，如图我们可以看到：

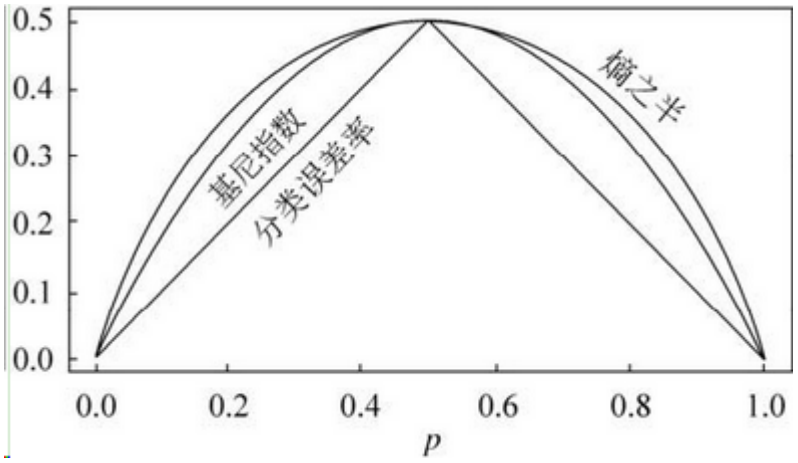

图 5.1 基尼系数和熵之半的曲线

基尼系数的曲线和熵之半的曲线整体保持逼近，仅仅在一些特殊位置会呈现较大的误差，例如上图的 45° 位置和 135° 位置。因此，基尼系数可以做为熵模型的一个近似替代。我们只需要每次对特征分类时只选择二分类，这样每个节点下都少于两个子节点，便能产生一颗二叉树。

对生成的二产树需要进行剪枝的同时还要考虑如何选择最优子树时，剪枝的标准是要使损失函数值变为最小。由于决策树的自身缺陷，过拟合的情况常常发生，导致泛化能力差，所以我们用类似与在线性回归中的正则化来给其进行剪枝。采用后剪枝法，即先生成决策树，然后产生所有剪枝后的 CART 树，然后使用交叉验证检验剪枝的效果，选择泛化能力最好的剪枝策略。

设正则化参数为α，训练数据的预测误差为$C(T_t)$，子树 T 的叶子节点个数是T_t。当$\alpha=0$时，我们视为不需正则化，则此时的 CART 树已经是最优的子树。当$\alpha=\infty$时，正则化程度已经最高，此时由原始的生成 CART 树的根节点组成的单节点树为最优子树。通常α越小，剪枝力度越小，相比于原来的树新生成的最优子树就越偏大。对于固定的 α，一定存在使得损失函数$C_\alpha(T_t)$最小的唯一子树。对于位于节点 t 的任意一颗子树T_t，如果没有剪枝，损失函数是：

$$C_\alpha(T_t)=C(T_t)+\alpha|T_t| \qquad \text{式(5-5)}$$

如果将其剪掉，仅保留根节点，损失函数是：

$$C_\alpha(T)=C(T)+\alpha \qquad \text{式(5-6)}$$

当 $\alpha=0$ 或 α 很小时，$C_\alpha(T_t)<C_\alpha(T)$，当 α 的值逐渐增大，会出现 $C_\alpha(T_t)=C_\alpha(T)$，若 α 继续增大，$C_\alpha(T_t)>C_\alpha(T)$。当出现 $C_\alpha(T_t)=C_\alpha(T)$ 时，α 满足：

$$\alpha=\frac{C(T)-C(T_t)}{T_t-1} \qquad \text{式(5-7)}$$

T_t 和 T 损失函数值是相等的，$C_\alpha(T_t)=C_\alpha(T)$。但是由于 T_t 节点更多，所以需要对子树 T_t 剪枝处理，生成新的叶子节点 T。

算法输入训练集，基尼系数的临界值，样本数目临界值，输出的是决策树 T。从根节点开始判断特征。用训练的子集逐步递归来建立决策树 CART。不断地分裂，当基尼系数的临界值大于实际的基尼系数、当样本数目临界值大于实际样本数目时便停止。不然，分裂直至可以选择最小的基尼系数。

最后预测时，样本在哪一个节点中落得的概率最大，则样本取此节点中的类别。

5.1.4 随机森林算法

在 Bagging 集成算法的基础上，我们进行改进，并使用 CART 决策树为最基本的分类器，这样便产生了最著名的分类算法之一：随机森林算法。

(1) 利用 Bootstrap 抽样方法任意地抽取 n 个样本。

(2) 在确定的所有特征中任意地选择 K 个特征，考虑如何划分特征使之效果最佳。

(3) 重复以上两步 m 次，建立 m 棵 CART 决策树。

(4) 这 m 个 CART 形成随机森林，最后类似于决策树，哪个类别的的频数大则选择此类别。

5.2 朴素贝叶斯分类

贝叶斯分类算法是依据贝叶斯定理为理论基础的一类分类算法。其中最简单最常用的朴素贝叶斯分类朴素在此处是一种强假设，指某件事情的各个特征之间是同等重要并且彼此无关的。

5.2.1 算法原理

由贝叶斯公式：

$$P(A|B)=\frac{P(B|A)}{P(B)}\times P(A) \qquad \text{式(5-8)}$$

我们可以进行分类：

设一个训练样本集 X，X={$x_1,x_2,x_3,...,x_n$}，对 i=1，2，3….n，其中每一个训练样本x_i有 j 个特征使得$x_i=(a_1,a_2,...,a_j)$。同时，这个训练样本可以被分成 k 个类别，分别为$C_1,C_2,...,C_k$。取一个独立于训练样本的测试样本y_1，只需要去求类别的后验概率，即哪种类别的后验概率最大y_1便被归为此种类别。计算y_1后验概率的表达式为：

$$\begin{aligned}P(C_i|y_1)&=\frac{P(y_1|C_i)}{P(y_1)}\times P(C_i)\\&=\frac{P(a_1,a_2,...,a_n|C_i)}{P(y_1)}\times P(C_i)\end{aligned} \qquad \text{式(5-9)}$$

根据我们给出的“朴素”条件，即事件的各个特征之间相互独立且同等重要，我们可以得到：

$$P(C_i|y_1)=\frac{\prod_n P(a_n|C_i)\times P(C_i)}{P(y_1)} \qquad \text{式(5-10)}$$

在上式中，对于不同的类C_i，分母的概率$P(y_1)$相同，即当$P(C_i|y_1)$最大，此时分子$\prod_n P(a_n|C_i)\times P(C_i)$最大。对于$P(a_n|C_i)$与$P(C_i)$我们利用训练样本集进行求解：

(1) 特征值是离散的

此时我们可以利用古典概率来求解，使用频率来拟替概率。当某个类别中不存在样本时，作 Laplace 平滑处理，当属于C_{i_1}分类中不为 0 时，分子分母都加 1，当属于C_{i_2}分类中为 0 时，分子变为 1，分母加上分类个数即可。

(2) 特征值是连续的

把样本中的每个特征看作为正态分布，这样我们就可以通过训练样本的集合得到正态分布的均值 μ_1 与标准差 σ_1。代入正态分布的标准公式中，得到了函数的高斯密度函数：

$$p(x)=\frac{1}{\sqrt{2\pi}\times\sigma_1}\times e^{-\frac{(x-\mu_1)^2}{2\sigma_1^2}} \qquad 式(5\text{-}11)$$

从而使用概率密度函数 p(x)来代替概率。

5.2.2 半朴素贝叶斯算法

由于“朴素”条件在现实生活中极为苛刻，各个特征之间很难是同等重要并且相互无关的。因此人们决定将特征独立的条件宽松化，形成了一种新的算法，被称为半朴素贝叶斯分类。其基本思想是仅仅尝试考虑一部分强的特征之间的依赖关系。常见的策略是“独依赖估计”，即指一个特征在分类外最多仅对其他的一个特征产生依赖性：

$$P(C_i \mid y_1) \propto P(C_i)\prod_{i=1}^{n} P(a_i \mid C_i, pa_i) \qquad 式(5\text{-}12)$$

pa_i 被称为父特征，是 a_i 所独依赖的一种另外的特征。假若 pa_i 已知，我们便可以估计 $P(a_i \mid C_i, pa_i)$。特征依赖于同一特征，并通过交叉验证来选择被依赖的特征，形成了一种 SPODE 方法。下图是朴素贝叶斯和半朴素贝叶斯之间的依赖关系。

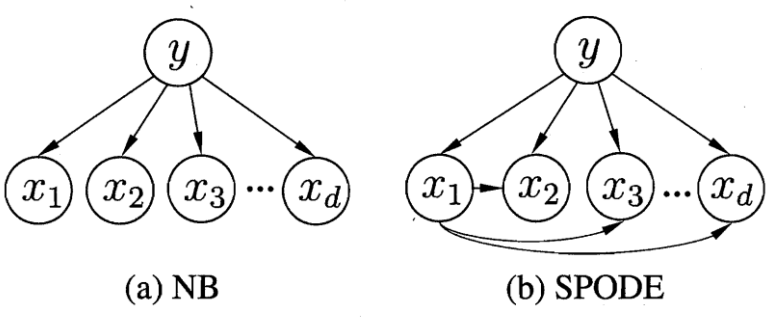

图 5.2 朴素和半朴素贝叶斯分类特征之间的关系

六、交叉验证

6.1 分类器性能评估指标

评估新预测变量的质量或者它的性能非常重要。为此，我们需要考虑以下两个问题。首先，应该使用哪些指标来衡量预测变量的质量？第二，应该采用什么方法来计算指标？下面，我们将解决这两个问题。

在文献中，下面四个指标经常被用于评价一个预测其的质量好坏：(1)准确性(Acc)；(2)稳定性(MCC);(3)灵敏性(Sn);(4)特异性(Sp)。但是他们的公式是直接取自数学书的，不够直观以至于难以被大多数生物科学家理解。幸运的是，在周研究信号肽时使用的引进的符号中，四个指标可以用如下更加直观的方式表达：

$$\begin{cases} Sn = 1 - \dfrac{N_-^+}{N^+} \\ Sp = 1 - \dfrac{N_+^-}{N^-} \\ Acc = \Lambda = 1 - \dfrac{N_-^+ + N_+^-}{N^+ + N^-} \\ Mcc = \dfrac{1 - (\dfrac{N_-^+ + N_+^-}{N^+ + N^-})}{\sqrt{(1 + \dfrac{N_+^- - N_-^+}{N^+})(1 + \dfrac{N_-^+ - N_+^-}{N^-})}} \end{cases} \quad \text{式(6-1)}$$

N^+ 代表研究的正样本数，N_-^+ 代表被错误预测为负样本的正样本数，N^- 代表研究的负样本数，N_+^- 代表被错误预测为正样本的负样本数。

上述指标表达式的出现，使得准确性、稳定性、灵敏性、特异性逐渐变得清晰明了，并在众多不同领域的不同研究中得到广泛应用。然而，它很有启发性地指出，随着测序分析研究进入了一个更深的层次，大量增多的多重标签测序样本正出现在系统生物学和药物学中。为了处理这种多重标签的系统，一组更加成熟的指标是目前迫切需要的。

下面三种不同的交叉验证方法通常被用于检验一个预测器的性能：(1)独立数据集测试;(2) K-Fold 交叉验证;(3)刀切法(Jackknife test)。独立数据集测试会在训练的过程中出现过拟合的问题，且划分为训练集和测试集时会浪费掉一部分的数据，无法使模型达到最优(数据决定了程性能上限，模型与算法会逼近这个上限。但是

我们又不能划分测试集，因为需要验证网络泛化性能，因此可以采用 K-Fold 交叉验证。

6.2 K-Fold 交叉验证算法

(1) 将含有 m 个样本的训练集 S 划分为 k 个不重合的独立子集，则每一个子集含有的样本数目有 $\frac{m}{k}$ 个，相应的，每一个不重合的子集称作 $\{S_1, S_2, ..., S_k\}$。

(2) 在训练子集中选择出 k-1 个 $\{S_1, S_2, S_{j-1}, S_{j+1}, ..., S_k\}$，再从模型中取一个 M_i，使用这 k-1 个子集训练 M_i 后，有假设函数 h_{ij}。使用没有被选择的 S_j 做测试，得到经验错误 $\varepsilon_{S_j}(h_{ij})$。

(3) 每一个 S_j 都会有一个经验错误，从而会求出 k 个经验错误，M_i 的经验错误是这 k 个经验错误和的平均值。

(4) 选出平均经验错误率最小的 M_i，然后使用训练集 S 做训练，得到最后的 h_i。

通过上述 1，2，3 步进行模型性能的测试，取平均值作为某个模型的性能指标。根据性能指标来挑选出最优模型，再进行上述第 4 步重新进行训练，获得最终模型。 然而数据总量较小时，其他方法无法继续提升性能，可以尝试使用 K-Fold 交叉验证。当数据量很大时，训练成本也要扩大 K 倍，就需要谨慎选择。

6.3 Jackknife 刀切法

同时，刀切法被验证是目前较为严谨也是较为客观的一种方法。因此，它已经被广泛地被研究者意识到并逐步应用到分析不同预测器的质量上。鉴于此，我们也一并说明刀切法的基本原理和具体步骤：

Jackknife 方法利用系统的划分数据集的办法来推测总体样本估计量的一些性质。起初它被用来估计偏差，随后研究者证明了运用它来估计估计量的方差也有很好的效果。其本质是将样本视为总体，在“总体”中不放回地抽取一些“样本”来做统计分析。

假设我们有随机样本 $S = (x_1, x_2, ..., x_n)$，定义第 i 个 Jackknife 样本为丢掉第 i 个样本后的剩余样本即：

$$x_{(-i)} = (x_1, ..., x_{i-1}, x_{i+1}, ..., x_n) \quad \text{式(6-2)}$$

$X_{(-i)}$ 被称为 Jackknife 伪值。

用上述方法产生的 Jackknife 样本集彼此区别非常小，两个不同的样本之间只有两个单独的原始样本不同。所以这种方法得到的未知参数的估计量都是偏性小或无偏性的。

其数学原理如下：

设 T(x)是基于样本 $S = (x_1, x_2, ..., x_n)$ 的关于 $g(\theta)$ 的估计量且满足：

$$E_\theta T(S) = g(\theta) + O(\frac{1}{n}) \quad \text{式(6-3)}$$

则可以构造：

$$T_J(S) = nT(S) - \frac{n-1}{n}\sum_{i=1}^{n} T(x_{(-i)}) \quad \text{式(6-4)}$$

满足：

$$E_\theta T_J(S) = g(\theta) + O(\frac{1}{n^2}) \quad \text{式(6-5)}$$

且方差不会增大。

偏差的 Jackknife 估计：

如果 θ 为一个平滑的估计量，则 $\theta_{(i)} = t(F_{n-1}(x_{(i)}))$，以及偏差的 Jackknife 估计为：

$$bias = (n-1)(\overline{\theta_{(\cdot)}} - \theta), \quad \text{式(6-6)}$$

其中 $\overline{\theta_{(\cdot)}} = \frac{1}{n}\sum_{i=1}^{n} \theta_{(i)}$.

$$\begin{aligned}&\mathrm{E}[\theta_{(\mathrm{i})}-\theta]=\mathrm{E}[\theta_{(\mathrm{i})}-\theta]-\mathrm{E}[\theta-\theta]\\&=\mathrm{bias}(\theta_{(\mathrm{i})})-\mathrm{bias}(\theta)\\&=-\frac{\sigma^2}{\mathrm{n}-1}-(-\frac{\sigma^2}{\mathrm{n}})\\&=\frac{\mathrm{bias}(\theta)}{\mathrm{n}-1}\end{aligned}$$ 式(6-7)

标准差的 Jackknife 估计：

记 $\theta_{(\mathrm{i})}=\frac{\mathrm{n}\bar{\mathrm{x}}-\mathrm{x}_{\mathrm{i}}}{\mathrm{n}-1}$，则 $\overline{\theta_{(\cdot)}}=\frac{1}{\mathrm{n}}\theta_{(\mathrm{i})}=\theta,\theta_{(\mathrm{i})}-\overline{\theta_{(\cdot)}}=\frac{\bar{\mathrm{x}}-\mathrm{x}_{\mathrm{i}}}{\mathrm{n}-1}$。因此有

$$\mathrm{se}=\sqrt{\mathrm{Var}(\theta)}$$ 式(6-8)

对平滑的统计量θ，其标准差的 Jackknife 估计定义为：

$$\mathrm{se}=\sqrt{\frac{\mathrm{n}-1}{\mathrm{n}}\sum_{\mathrm{i}=1}^{\mathrm{n}}(\theta_{(\mathrm{i})}-\overline{\theta_{(\cdot)}})^2}$$ 式(6-9)

比如当θ为总体均值时，$\theta=\bar{\mathrm{x}}$，其方差估计为：

$$\mathrm{Var}(\theta)=\frac{\sigma^2}{\mathrm{n}}=\frac{1}{\mathrm{n}(\mathrm{n}-1)}\sum_{\mathrm{i}=1}^{\mathrm{n}}(\mathrm{x}_{\mathrm{i}}-\bar{\mathrm{x}})^2$$ 式(6-10)

七、结果讨论

按照前面叙述的方法步骤，依次进行特征提取、特征选择、调整不平衡数据以及利用不同分类器训练，结合十折交叉验证，我们得到了以下的结果：

分类器	ACC(%)	Mcc(%)	Sn(%)	Sp(%)
SVM	92.0%	84.7%	98.7%	86.8%
朴素贝叶斯	80.3%	60.9%	83.1%	78.0%
随机森林	85.9%	71.9%	83.3%	88.7%

表 7.1 分类器测试结果

通过三种不同的分类器训练样本，我们可以发现 SVM 和集成袋装树的准确度较高，朴素贝叶斯的分类结果略低，这说明不同的预测方法对预测结果对有一定的影响，同时也都具有一定的价值。另外，文中处理不平衡数据利用的是 SMOTE 算法，对准确度也起到了一定的正向作用。使用 PCA 降维极大地稀疏了数据的特征，既保持了较高的准确率，也能对于训练模型的时间缩短起到了非常显著的效果。这大大地体现了机器学习提取信息和预测位点上的优越性，开辟了一条利用数学与计算机而革新生物学的道路。

八、结束语

本文虽然在研究 DNA 超敏位点预测中，阐述了一定的研究基本思想和方法，并进行了最终的结果评估，取得了不错的成效，但是由于自身水平有限时间紧迫，还有很多工作尚未完成，需要进一步的完善与改进：

1）对于 DNA 序列特征信息提取有众多的独特见解，本文的重点在于利于物化性质来完成这项工作，随着生化学的日益发展，数据的爆发式膨胀，需要进一步思考新的特征信息提取方法，来应对更深入的进展与挑战。

2）对于不平衡数据的处理，由于时间有限，在实际测试中只使用了 SMOTE 算法，而未采用 BL-SMOTE 算法，其数据处理方法理论上应显著提高训练模型的诊断能力，但具体表现仍有待进一步改进提升。

3）在预测器模型中，我们在文中使用了三种最基础最通用的模型，基于 DNA 序列我们对于 DNA 超敏位点展开了研究，同时，这种思想与理论实践也可以推广泛用到其他生物领域中，诸如对 RNA 和蛋白质的功能进行探索。

九、参考文献

[1]邹国英. 基于机器学习方法的 DNA 位点预测研究[D].景德镇陶瓷大学，2016.

[2]杨润涛. 分子生物学系统建模及蛋白质功能预测相关问题研究[D].山东大学，2016.

[3]魏志森. 蛋白质相互作用位点预测方法研究[D].南京理工大学，2016.

[4]刘全亚. 蛋白质突变位点数据库的构建及位点预测研究[D].安徽大学，2019.

[5]丛瑞达. 基于 DNase-Seq 的转录因子 DNA 蛋白结合位点识别方法研究[D].哈尔滨工程大学，2019.

[6]惠梦娟. 基于序列特征的蛋白质功能类型预测器研究[D].景德镇陶瓷大学，2016.

[7]王鹏. 深度学习框架下 DNA 位点的预测研究[D].景德镇陶瓷大学，2018.

[8]杨骥.基于序列与结构特征结合的蛋白质与 DNA 绑定位点预测[J].计算机与现代化，2016(01)：20-25.

[9]Zhang Shengli，Yu Qianhao，He Haoran，Zhu Fu，Wu Panjing，Gu Lingzhi，Jiang Sijie. iDHS-DSAMS： Identifying DNase I hypersensitive sites based on the dinucleotide property matrix and ensemble bagged tree.[J]. Genomics，2020，112(2).

[10]Sun， J.， Wu， Q.， Shen， D. *et al.* TSLRF： Two-Stage Algorithm Based on Least Angle Regression and Random Forest in genome-wide association studies. *Sci Rep* **9**， 18034 (2019). https：//doi.org/10.1038/s41598-019-54519-x

[11]Pengmian Feng，Hui Yang，Hui Ding，Hao Lin，Wei Chen，Kuo-Chen Chou. iDNA6mA-PseKNC： Identifying DNA N 6 -methyladenosine sites by incorporating nucleotide physicochemical properties into PseKNC[J]. Genomics，2018.

[12]Liang Yunyun，Zhang Shengli. Identifying DNase I hypersensitive sites using multi-features fusion and F-score features selection via Chou's 5-steps rule.[J]. Biophysical chemistry，2019，253.

十、致谢

感谢我的指导老师张胜利老师。张胜利老师于我而言亦师亦友，学识渊博却又充满人格魅力。在撰写毕业论文的时候，从框架思路、文献选读、格式规范等等不同的方面张胜利老师都不厌其烦地解答我的困惑，给予合适的思路，并提供指导性的意见。在接下里的学习生涯规划方面给予了莫大的帮助，使我终身受益。

西电是一所正直朴素却又有深度的学校，在这四年的相伴里，我认识了众多的老师和同学，在我有困难的时候总是主动出手相助，带给我欢笑、帮助和祝福，给我留下值得珍藏一生的回忆，也让我明白作为大学生的使命与责任。

在学习的路上继续前行，不禁感激我的母校西安电子科技大学，感谢在学校里相遇的老师和同学们，在我遇到困难时主动伸出援手。莫道秋江离别难，舟船明日是长安！

附录 A

	AA/TT	AC/GT	AG/CT	AT	CA/TG	CC/GG	CG	GA/TC	GC	TA
Entropy	-21.3	-22.4	-21	-20.4	-22.7	-19.9	-27.2	-2.2	-24.4	-21.3
Enthalpy	-7.6	-8.4	-7.8	-7.2	-8.5	-8	-10.6	-8.2	-9.8	-7.2
Energy	-1	-1.44	-1.28	-0.88	-1.45	-1.84	-2.17	-1.3	-2.24	-0.58
Rise	3.25	3.24	3.32	3.21	3.37	3.36	3.29	3.3	3.27	3.39
Shift	0.01	-0.02	-0.02	0	0.01	0.03	0	-0.01	0	0
Slide	-0.18	-0.59	-0.22	-0.68	0.48	-0.17	0.44	-0.05	-0.19	0.04
Twist	35.02	31.53	32.29	30.72	35.43	33.54	33.67	35.67	34.07	36.94
Tilt	-1.26	0.33	-1.66	0	0.14	-0.77	0	1.44	0	0
Roll	1.05	2.01	3.6	0.61	5.6	4.68	6.02	2.44	1.7	3.5
F-rise	21.34	21.98	17.48	24.79	14.51	14.25	14.66	18.41	17.31	14.24
F-shift	6.24	2.91	2.8	4.66	2.88	2.67	3.02	3.58	2.66	4.11
F-slide	6.69	6.8	3.47	9.61	2	2.99	2.71	4.27	4.21	1.85
F-twist	0.07	0.06	0.05	0.07	0.05	0.06	0.05	0.06	0.06	0.05
F-tilt	0.08	0.07	0.06	0.1	0.06	0.06	0.06	0.07	0.07	0.07
F-roll	0.04	0.06	0.04	0.05	0.04	0.04	0.04	0.05	0.05	0.03

表 2.2 二核苷酸物化性质值

附录 B

Python 源代码：

```
#特征提取
seq = []
line = ''
m = open('C:/Users/yang/Desktop/Code/Data-S1.txt', 'r')
for line in m.readlines():
    if line.startswith('A'):
        seq.append(line.strip())
    if line.startswith('C'):
        seq.append(line.strip())
    if line.startswith('G'):
        seq.append(line.strip())
    if line.startswith('T'):
        seq.append(line.strip())
value_na = ['AA', 'TT', 'AT', 'AG', 'CT', 'AC', 'GT', 'TA', 'TG', 'CA', 'TC', 'GA', 'GG', 'CC', 'GC',
'CG']
Pro = np.array(excel2m('C:/Users/yang/Desktop/Code/pro.xlsx'))
Pro_Nm = np.transpose(MinMaxScaler().fit_transform(np.transpose(Pro)))
matrix = []
for t in range(0, len(seq)):
    value_matrix = np.zeros((len(seq[t]) - 1, Pro_Nm.shape[0]))
    words = []
    words_1 = []
    for word in seq[t]:
        words += word
    del (words[-1])
    for word1 in seq[t]:
        words_1 += word1
```

```
        del (words_1[0])
        word_sum = []
        for i in range(len(words)):
            word_sum.append(words[i] + words_1[i])
        for i in range(len(word_sum)):
            for ii in range(len(value_na)):
                if word_sum[i] == value_na[ii]:
                    for x in range(Pro_Nm.shape[0]):
                        value_matrix[i][x] = Pro_Nm[x][ii]
        matrix.append(value_matrix)
result = np.zeros((15, 15))
G = np.zeros((225, 5))
feature = np.zeros((1017, 1125))
for i in range(1017):
        value_matrix = matrix[i]
        for g in range(5):
            for s in range(15):
                value_s = value_matrix[:, s]
                csg1 = value_matrix[0:len(seq[i]) - 1 - g - 1, s]
                csg2 = value_matrix[1 + g:len(seq[i]), s]
                for t in range(15):
                    value_t = value_matrix[:, t]
                    denominator = sum((value_s - value_s.mean() * (value_t - value_t.mean())))
                    denominator = denominator / (len(seq[i]) - 1 - 1)
                    ctg1 = value_matrix[0:len(seq[i]) - 1 - g - 1, t]
                    ctg2 = value_matrix[1 + g:len(seq[i]), t]
                    numerator = sum(np.multiply((csg1 - csg2), (ctg1 - ctg2)))
                    numerator = numerator / (2 * (len(seq[i]) - 1 - g))
                    result[s, t] = numerator / denominator
```

```
        x = 0
        while x < result.shape[0] * result.shape[1]:
            for cols in range(result.shape[1]):
                for rows in range(result.shape[0]):
                    G[x, g] = result[rows][cols]
                    x += 1
x = 0
while x < G.shape[0] * G.shape[1]:
        for cols in range(G.shape[1]):
            for rows in range(G.shape[0]):
                G1 = np.transpose(G)
                feature[i, x] = G1[cols][rows]
                x += 1
```