机器分类算法

项目报告

学　　院：国家卓越工程师学院

班　　级：明月科创 1 班

姓　　名：莫湘渝

学　　号：20232373

指导老师：徐建文

摘要

本项目聚焦于机器学习中的二分类问题，围绕支持向量机（Support Vector Machine, SVM）算法展开研究，探讨其在不同数据结构下的分类性能及模型改进的效果，并与经典的 Logistic 回归模型进行对比分析。通过在 Raisin 数据集上构造二维与多维特征空间，基于标准 SVM、Logistic 回归以及改进 SVM（Clip-DCD 算法，含线性与 RBF 核）模型进行建模和评估。实验结果表明，在样本区分度弱的情形下，Logistic 回归性能更为稳定；而在样本区分度明显时，SVM 具有更优的泛化能力。此外，利用核函数扩展的非线性 SVM 以及 Clip-DCD 算法在弱可分场景中表现出显著的性能提升。项目使用 MATLAB 完成所有建模与可视化工作，并实现了 CNN 作为对照模型。最后，对相关 SVM 改进文献中的正则项与损失项设计进行调研与总结。

关键词：支持向量机、Logistic 回归、Clip-DCD、非线性核函数、RBF 核、正则化、二分类、机器学习、MATLAB、Raisin 数据集

1 项目背景任务

1.1 背景介绍

人工智能（Artificial Intelligence, AI）这一概念最早由约翰·麦卡锡等人在 1956 年达特茅斯会议上提出，其核心目标是通过计算机模拟人类的思维过程和智能行为，使得机器具备类似人类的感知、推理、学习和决策能力。随着技术的不断进步，人工智能已广泛应用于机器翻译、智能控制、图像识别、语音识别、自然语言处理、游戏对弈等多个领域，成为推动科技发展的重要引擎。

在人工智能的发展过程中，机器学习（Machine Learning）作为其核心分支之一，得到了快速发展。机器学习的思想最早可以追溯至 20 世纪 50 年代提出的

感知机模型，其目标在于赋予机器类似人类的学习能力。机器学习的基本思想是基于样本数据对某种模型进行训练，从而建立输入与输出之间的映射关系。常见的机器学习框架，如正则化学习理论，通常包括三个关键组成部分：学习算法（或称学习器）、损失函数（用于度量预测误差）和正则项（用于防止模型过拟合）。通过最优化方法对这些组成部分进行协同求解，最终得到泛化能力较强的模型。

支持向量机（Support Vector Machine, SVM）是机器学习中的一种重要监督学习方法，最初由 Cortes 与 Vapnik 于 1995 年提出，专用于处理二分类问题[^1]。SVM 具有坚实的统计学习理论基础，其核心思想是寻找一个最优超平面，将不同类别的样本尽可能地分隔开来。与许多机器学习方法相比，SVM 不仅在小样本、高维度、非线性问题中表现出色，而且模型结构简洁，几何意义明确，易于实现，因此在图像识别、文本分类、生物信息学等应用场景中被广泛采用。

除了 SVM，Logistic 回归也是一种经典的二分类方法，源于广义线性模型（GLM）体系。Logistic 回归通过对因变量取对数几率变换，将分类问题转化为线性建模问题。尽管其假设相对简单，但在许多实际问题中依然有着良好的表现。与 SVM 类似，Logistic 回归研究的是一组协变量 X1,X2,...,Xp 对二元响应变量 Y 的影响，因此两者在实际分类任务中经常被同时采用和比较。理解这两种模型的差异性对于模型选择具有重要意义。

值得一提的是，SVM 的发展并未止步于最初的线性可分硬间隔模型。考虑到现实数据往往存在噪声、重叠等问题，研究者提出了软间隔 SVM，引入松弛变量以提高模型的容错能力。此外，为解决非线性可分问题，核函数方法应运而生，通过在高维特征空间中实现数据的线性可分，进而实现非线性分类。近年来，针对 SVM 中损失函数与正则项的不同设定，学术界又提出了多种变种模型和求解算法，不断扩展其理论深度与应用广度。

综上所述，SVM 作为机器学习中的重要模型，既具有丰富的理论内涵，也具有较强的实用价值。

1.2 项目任务

在本次项目中，需要寻找合适的数据，并在该样本数据基础上分别利用经典 SVM 模型与 Logistic 模型进行统计建模，同时对比两者的分类效果；此外还需

要总结并实现部分改进版本的 SVM 算法，分析其预测效果。具体而言可细分为如下任务：

任务 1：数据生成与经典模型建模分析

在本项目中，选用的数据为 kaggle 上开源的训练数据——关于两种葡萄籽分类依据的数据。首先读取 Raisin_Dataset.xlsx 数据集，并选取 6 个关键形状特征作中两个较为显性的数据作为特征变量，对其进行标准化预处理。通过 cvpartition 将总样本划分为训练集和测试集。随后，我们在选定的两个特征维度（如 ConvexArea 与 Extent）下，构建二维特征子集以便可视化分析，同时保留六维数据用于更高维度的分类效果比较。

在二维特征空间中，我们基于训练集分别采用经典的硬间隔支持向量机（SVM，线性核及 RBF 核）与带 L1 正则化的逻辑回归（Logistic Regression）进行建模，并在测试集上评估其分类效果。通过准确率、ROC 曲线、AUC 值以及混淆矩阵对比分析不同模型的性能表现。此外，为进一步展示模型的决策边界，还通过图形化方式呈现各分类器在二维与主成分映射空间中的分类区域。

任务 2：改进版本 SVM 模型的实现与对比

在原始 SVM 模型的基础上，项目中实现了一种改进的 SVM 变种算法——Clip-DCD（Clipped Dual Coordinate Descent）算法，包括线性核与 RBF 核两个版本。该方法通过对标准 SVM 优化过程中的坐标下降策略进行剪枝处理，提高了求解效率与泛化能力。基于相同训练集与测试集环境，使用 clipdcd_svm 与 clipdcd_svm_rbf 两个实现对样本进行分类建模，并将其与传统硬间隔 SVM 模型在分类准确率与 AUC 表现上的差异进行对比评估，结果显示 Clip-DCD 模型在某些维度上具有更强的分类能力与鲁棒性。

任务 3：SVM 改进算法的文献研究与总结

在文献调研的基础上，项目聚焦于基于正则化框架下 SVM 模型的扩展与改进方向。以 Clip-DCD 方法为例，其本质仍属于二次规划问题的凸优化任务。模型创新之处在于其损失项采用了硬间隔下的 hinge 损失函数，正则项保持 L2 范式，而优化求解则使用了一种经过剪枝的坐标下降策略，有效降低了冗余变量

更新次数。

此外，项目中还引入了一个简化版的人工神经网络模型（Simple CNN），作为一种非线性模型基准，用于对比基于特征映射的传统方法与基于参数学习的深度方法之间的效果差异。该 CNN 网络包括一个一维卷积核与全连接输出层，通过反向传播实现训练，并集成进分类任务的可视化流程中。

2 实现过程

2.1 数据获取

本项目中涉及到的样本数据有两种来源。一种是通过 matlab 生成的二维正态总体数据，一种是 kaggle 上开源的实际数据。在上述数据生成的基础上，本文选取支持向量机（SVM）与逻辑回归（Logistic 回归）两类经典的二分类模型对模拟数据进行建模与分析。SVM 模型通过在特征空间中寻找最优分类超平面以最大化类别间的间隔，从而实现对数据的有效分类；而逻辑回归模型则通过建立响应变量 Y 与协变量 X1,X2 之间的对数几率关系，以概率形式刻画样本属于某一类别的可能性，并以此实现分类任务。这两类模型均能够在解释协变量与响应变量关系的同时，用于预测新的观测数据的类别，便于后续进行性能对比分析。

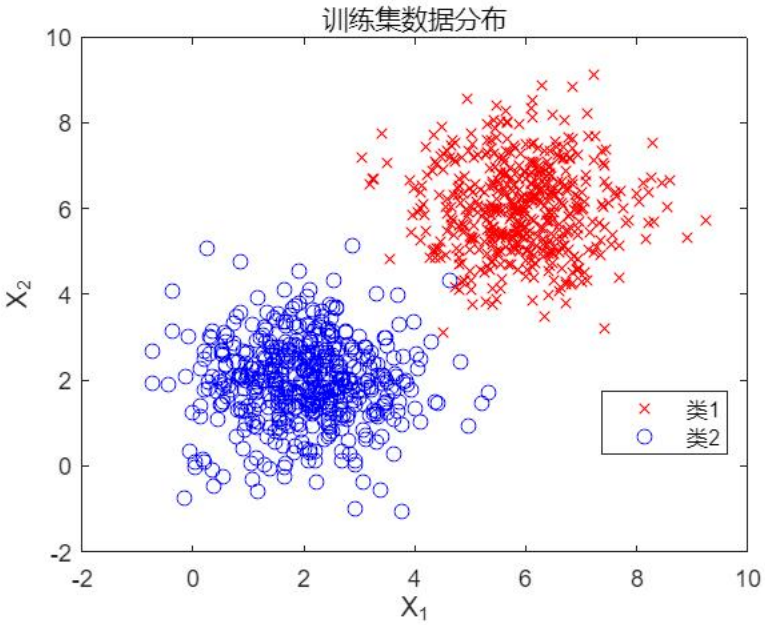
第一类数据：具有较为规率的分布，并且边界较为清晰，可调节重叠程度，主要用于初步的 SVM 硬件隔和 logistic 比较。在数据生成过程中，首先固定随机种子以保证可重复性。模拟了一个二维二分类问题，其中：

类别为 1 的样本从二维正态分布中生成，均值向量为 [2,2]，协方差矩阵为单位矩阵 I2，分别生成 500 个训练样本和 500 个测试样本。

类别为 2 的样本同样从二维正态分布中生成，均值向量为 [6,6]，协方差矩阵同为单位矩阵 I2，同样分别生成 500 个训练样本和 500 个测试样本。

```
% 类别1：均值[2,2]
mu1 = [2, 2];
sigma1 = eye(2);
X1_train = mvnrnd(mu1, sigma1, 500);
X1_test = mvnrnd(mu1, sigma1, 500);

% 类别2：均值[6,6]
mu2 = [6, 6];
sigma2 = eye(2);
X2_train = mvnrnd(mu2, sigma2, 500);
X2_test = mvnrnd(mu2, sigma2, 500);
```

随后，将两类数据在训练集和测试集内分别进行合并，构成最终的训练特征矩阵X_{train}与标签向量y_{train}，以及测试特征矩阵X_{test}与标签向量y_{test}。最终得到一个平衡的二分类数据集，每个类别在训练集和测试集中均包含 500 个样本，用于后续支持向量机（SVM）和逻辑回归模型的训练与性能评估。



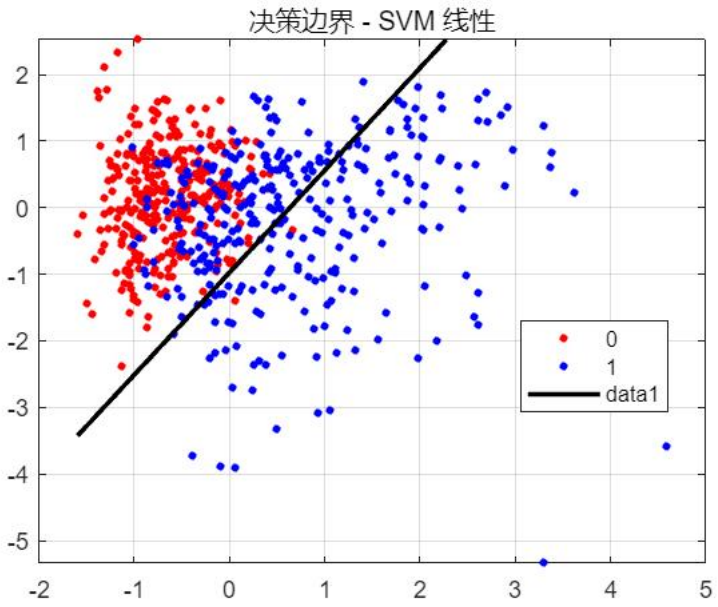
第二类数据，选用 kaggle 的开源数据，关于不同种类的葡萄籽外形数据。Raisin_Dataset.xlsx 数据集作为分类任务的真实数据来源，以便验证模型在实际情境下的适用性。数据集中有“Area”、“MajorAxisLength”、“MinorAxisLength”、“Eccentricity”、“ConvexArea”、“Extent”、“Perimeter”七个特征作为自变量，并将原始表格数据转换为矩阵形式进行后续分析。为了保证各特征具有可比性，对所有特征进行标准化处理，使其均值为 0，方差为 1。

```
%% 1. 读取并预处理数据
data = readtable('Raisin_Dataset.xlsx');
features = {'Area', 'MajorAxisLength', 'MinorAxisLength', ...
            'Eccentricity', 'ConvexArea', 'Extent', 'Perimeter'};
X = table2array(data(:, features));
Y = data.Class;
Ynum = grp2idx(Y) - 1;  % 转为 0 / 1
X = zscore(X);          % 标准化

cv = cvpartition(Ynum, 'HoldOut', 0.3);
Xtrain = X(training(cv), :);   Ytrain = Ynum(training(cv));
Xtest  = X(test(cv), :);       Ytest  = Ynum(test(cv));
```

随后，采用 70% 作为训练集，30% 作为测试集进行数据集划分，以保证模型训练和测试的相互独立性。

为了便于后续可视化分析，同时从已标准化后的七个特征中选择“ConvexArea”与“Extent”两个特征用于构建二维特征子集，以便在二维平面上直观展示模型在训练集与测试集上的分类效果，并用于可视化 SVM 与 Logistic 回归在实际数据集下的分类边界。

```
% 设置用于2维可视化的特征索引
xFeat = 5; yFeat = 6;  % 例如 ConvexArea & Extent
X2Dtrain = Xtrain(:, [xFeat, yFeat]);
X2Dtest  = Xtest(:, [xFeat, yFeat]);
```



数据的重叠程度较大，对于后面做 SVM 数据硬间隔有一定挑战。

2.2 硬件隔 SVM 模型

2.2.1 原理解析

硬间隔支持向量机是一种用于线性可分数据集的二分类模型，其核心目标是构造一条最优的分类超平面，将两类样本完全分离。所谓线性可分，指存在一条超平面（在二维情况下即为直线）可以将正负类样本无误地区分开来。虽然满足这一条件的超平面有无数条，但 SVM 的关键在于从中选取具有最大分类间隔（margin）的一条作为最优解。

该最大间隔的分类器在几何上表现为距离两类样本最近样本点（即“支持向量”）最远的分隔超平面。支持向量机的基本思想是将这一问题形式化为一个带不等式约束的凸优化问题。

设训练数据为$(x_i, y_i), i = 1, ..., n$，其中$x_i \in \mathbb{R}^p$ 表示第 i 个样本，$y_i \in \{-1, +1\}$表示其对应的类别标签。一个线性分类器可表示为超平面：

$$w^T x + b = 0$$

其中$w \in \mathbb{R}^p$是法向量，b 是偏置项。SVM 的目标是最大化两类样本到该超平面的最小几何间隔（margin）。已知支持向量到超平面的距离为：

$$d_i = \frac{|w^T x_i + b|}{\|w\|}$$

若分类器能正确分类所有样本，则对于任意样本 (x_i, y_i) 都应满足：

$$y_i(w^T x_i + b) \geq 1$$

结合几何意义，SVM 的优化目标即为最大化最小间隔，等价于最小化 $\frac{1}{2}\|w\|^2$，因此原始问题可表述为如下约束优化问题：

$$\min_{w,b} \ \frac{1}{2}\|w\|^2$$
$$\text{s.t. } y_i(w^T x_i + b) \geq 1, \ i = 1, ..., n$$

这是一个典型的二次规划问题，具有良好的凸性结构。为求解该问题，通常采用拉格朗日对偶方法，并引入 KKT（Karush-Kuhn-Tucker）条件进行求解。通过构造拉格朗日函数并求解其对偶问题，可以有效地识别出那些对间隔起决定作用的训练样本，即**支持向量**。这些样本在最终的分类函数中起到关键作用：

$$f(x) = \text{sgn}(w^T x + b)$$

一旦求得最优参数 w^*、b^*，即可得到最优的线性分类器。对于新的样本点 x，若 f(x) > 0，则预测类别为 +1，否则为 -1。

2.2.2 项目实现

在本项目中，我们假设协变量维度 p=2，因此分类器表示为二维空间中的一条直线，其学习过程即为在二维空间中寻找一条最大化两类间隔的分隔线。

MATLAB 中有相应函数实现硬件隔和软间隔的 SVM，在实现硬件隔 SVM 时，要求没有松弛变量（ξ_i），并且严格要求数据完全可分，内部求解依然是通过 **拉**格朗日对偶、KKT 条件、SMO 等方法完成，调用函数 fitcsvm 实现模型训练，其中 KernelFunction = 'linear'使用线性核，相当于求解$w^Tx+b=0$。

BoxConstraint = 1e6 对应软间隔 SVM 中的 C 参数：

$$\min_{w,b,\xi} \ \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i$$

$$\text{s.t.}\ y_i(w^Tx_i+b) \geq 1-\xi_i,\ \xi_i \geq 0$$

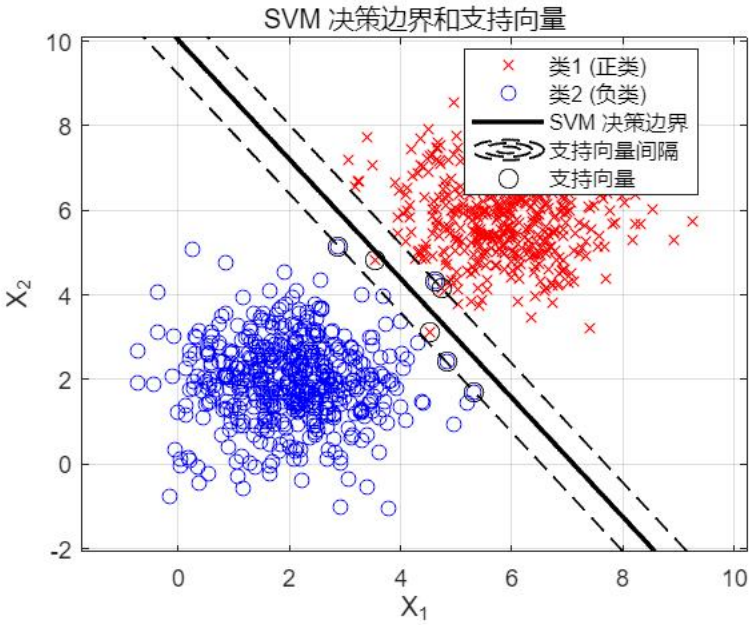
当 C（即 BoxConstraint）非常大时，会强迫 ξ_i 尽量接近 0，避免任何误分类。当 $C \to \infty$，等价于硬间隔 SVM。但是注意到若数据确实线性可分，使用 BoxConstraint = 1e6（或更大）是合适且等价于硬间隔 SVM。若数据不完全线性可分，过大的 BoxConstraint 会导致无法收敛，此时应使用适当的软间隔 (BoxConstraint = 1~10) 实际应用中。

```
%% 2. 硬间隔 SVM 模型（BoxConstraint设置很大）
svmModel = fitcsvm(X_train, y_train, ...
    'BoxConstraint', 1e6, ...
    'KernelFunction', 'linear');
```

```
% SVM 预测
y_pred_svm = predict(svmModel, X_test);
acc_svm = mean(y_pred_svm == y_test);
```

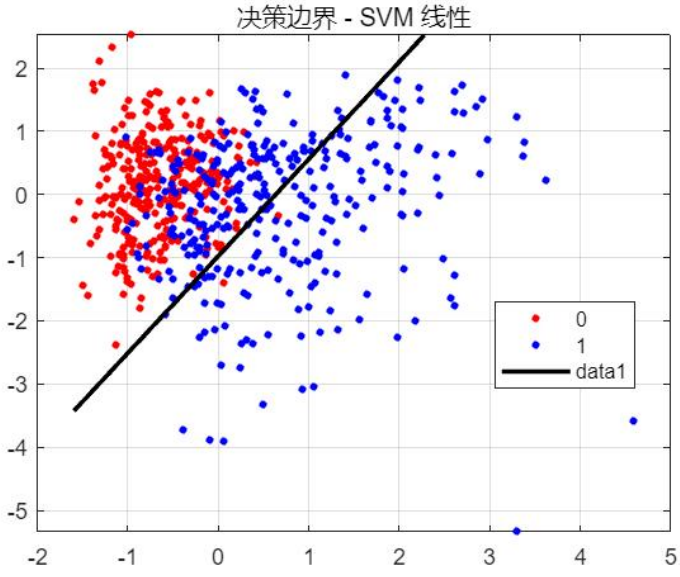
使用模拟生成的正态数据，满足完全可分，此时可以得到如下的训练效果：成功率几乎百分百。

```
SVM Accuracy: 99.80%
```



但是真是的数据往往不会特别规律，并且不会完全线性可分，当我使用真实数据进行硬件隔 SVM 训练准确率会大大下降，准确率在 60-70%左右跳动，效果并不好。并且此时硬件隔就失效了。这个约束$y_i(w^T x_i + b) \geq 1$ 就不可能同时满足，SVM 优化问题将无解或数值不稳定。下面能够得到结果，决策线图像的原因是我的 BoxConstraint 参数并没有设为无限大，此时就相当于使用了惩罚力度很大的软间隔 SVM。

```
% 2. SVM 硬间隔（线性核）
model_svm = fitcsvm(X2Dtrain, Ytrain, 'BoxConstraint', 1e6, 'KernelFunction','linear')
Ypred_svm = predict(model_svm, X2Dtest);
acc_svm = mean(Ypred_svm == Ytest);
fprintf('2D SVM (线性) 准确率: %.2f%%\n', acc_svm*100);
```



```
2D SVM (线性) 准确率: 74.44%
```

是否可以思考使用不同的核函数实现硬件隔 SVM 呢，fitcsvm 函数提供了使用不同核函数的接口，这里我们将'KernelFunction',后面设为 'rbf'，使用高斯核进行 SVM。

高斯核是一种非线性核函数，将数据映射到无限维的特征空间，使得在原空间中非线性可分的数据在核空间中线性可分。实现复杂决策边拟合。线性核仅在原空间寻找最优超平面，只能拟合线性可分问题，计算快，可解释性强。高斯核需调节核宽度和 BoxConstraint 参数，适合数据分布复杂或非线性边界情况；线性核适合数据线性可分或维度很高但样本少的场景。两者在 fitcsvm 中使用的求解器和底层优化框架一致，只是在计算核矩阵与决策边界的非线性形式上有所不同。

高斯核（Radial Basis Function, RBF）定义：

$$K(x_i, x_j) = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right)$$

其中$\|x_i - x_j\|^2$ 为两样本间欧几里得距离平方。σ为核宽度参数，控制“邻域范围”。在 fitcsvm 中 KernelScale = σ。

在此核空间中，SVM 实际仍在寻找一个超平面：

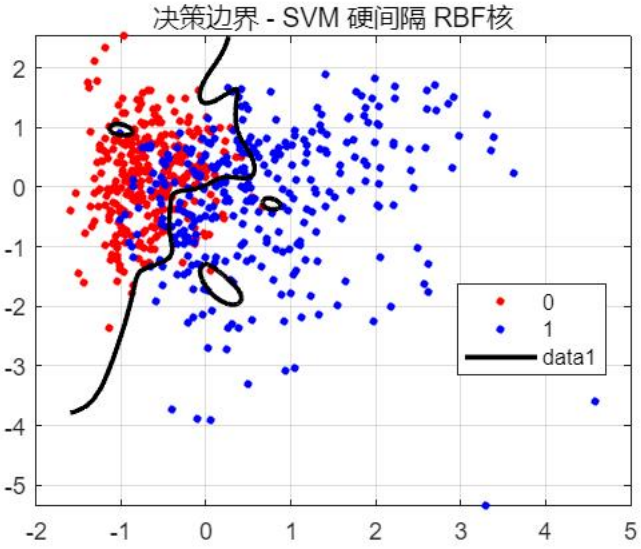
$$f(x) = \text{sign}\left(\sum_{i=1}^{n} \alpha_i y_i K(x_i, x) + b\right)$$

其中，α_i为拉格朗日乘子，$K(x_i, x)$为高斯核计算出的相似度。因为高斯核仅涉及训练样本与预测样本距离，而无需显式计算高维映射，避免了“维度灾难”。

```
%SVM 硬间隔（高斯核，宽度1）
model_svm_rbf = fitcsvm(X2Dtrain, Ytrain, 'BoxConstraint', 1e6, 'KernelFunction', 'rbf', 'KernelScale', 1);
Ypred_svm_rbf = predict(model_svm_rbf, X2Dtest);
acc_svm_rbf = mean(Ypred_svm_rbf == Ytest);
fprintf('2D SVM 硬间隔 (RBF核) 准确率: %.2f%%\n', acc_svm_rbf*100);
```

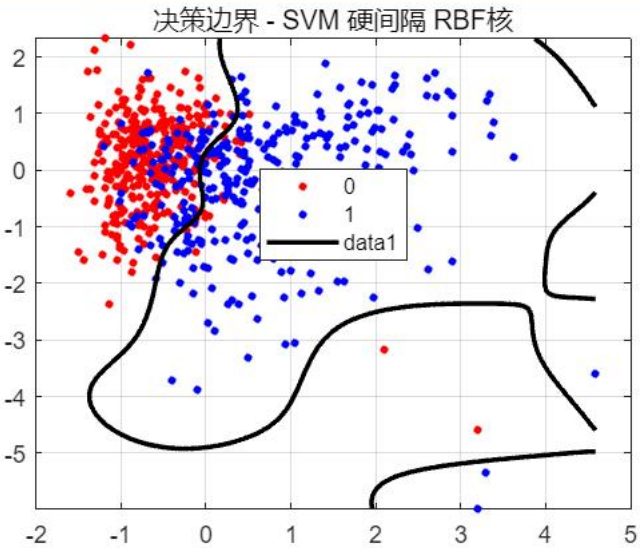
这里虽然设置了“硬间隔” (BoxConstraint = 1e6)，但由于使用了高斯核，

实际是在高维核空间中寻求无误分类超平面。若数据本身重叠（非线性可分），即便高斯核也无法严格无误分类，可能出现求解困难或过拟合，通常应使用有限的 BoxConstraint 实际控制容错。




```
2D SVM 硬间隔 (RBF核) 准确率: 70.37%
```

可以看到，高斯核在实际实现时，与线性核的分类效果较为接近。这与惩罚参数 C 的设置密切相关。当将惩罚参数提高到 C=5 时，模型在测试集上的准确率能够提升至 80% 左右，此时模型已处于软间隔支持向量机（soft margin SVM）状态。关于软间隔与硬间隔下参数对模型泛化能力和分类边界的影响，将在后续章节详细讨论。




```
2D SVM 硬间隔 (RBF核) 准确率: 83.70%
```

2.3 Logistic 回归

与 SVM 模型类似，Logistic 回归模型同样旨在研究协变量（X1,…,Xp）如何影响响应变量 yyy，从而进行分类预测。但与 SVM 基于最大间隔划分类别的思路不同，Logistic 回归作为 广义线性回归模型（GLM）的一种特殊情况，本质上仍是基于“回归”的思想，通过建立概率模型进行分类预测。

2.3.1 原理分析

①模型假设：

考虑二元响应变量 y∈{0,1}，并假定独立观测 n 次：

$$yi \sim B(1, \mu i), i = 1, \ldots, n$$

其均值和方差分别为：

$$E(y_i) = \mu_i,\ Var(y_i) = \mu_i(1 - \mu_i)$$

同时，协变量 $X_1, \ldots, X_p$对应 n 次观测，记其线性组合：

$$\eta_i = \beta_0 + x_{i1}\beta_1 + \cdots + x_{ip}\beta_p = x_i^T\beta,\ i = 1, \ldots, n$$

其中：

$$x_i = (1, x_{i1}, \ldots, x_{ip})^T,\ \beta = (\beta_0, \beta_1, \ldots, \beta_p)^T$$

针对二元分类问题，通常选用 Logistic 链接函数：

$$g(x) = \ln\frac{x}{1 - x},\ x \in (0,1)$$

该函数是 logit 函数，其反函数（sigmoid 函数）为：

$$g^{-1}(\eta_i) = \frac{1}{1 + e^{-\eta_i}}$$

因此，Logistic 回归模型可表达为：

$$\boxed{\mu_i = P(y_i = 1|x_i) = \frac{1}{1 + e^{-x_i^T\beta}}}$$

模型给出的是样本 xi 属于类别 1 的概率预测，通过概率阈值（通常取 0.5）实现分类。

在训练 Logistic 回归模型时，目标是估计参数向量 β 以拟合训练数据。对于每个观测值：

$$P(y_i|x_i) = \mu_i^{y_i}(1-\mu_i)^{1-y_i}$$

因此，观测值的对数似然函数为：

$$l(\beta) = \sum_{i=1}^{n}[y_i \ln \mu_i + (1-y_i)\ln(1-\mu_i)]$$

训练过程即为对 β 进行极大似然估计（MLE），最大化上述对数似然函数。与线性回归的最小二乘估计不同，Logistic 回归中对数似然函数无显式解，通常使用 Newton-Raphson 方法（IRLS）；拟牛顿法（BFGS）；梯度下降法（SGD）；等数值优化方法求解。训练完成后，得到最优参数 β^。对于任意测试样本 x，预测概率：

$$p = \frac{1}{1+e^{-x^T\hat{\beta}}}$$

若 p>0.5，预测 y=1；若 p≤0.5，预测 y=0。即可完成分类。

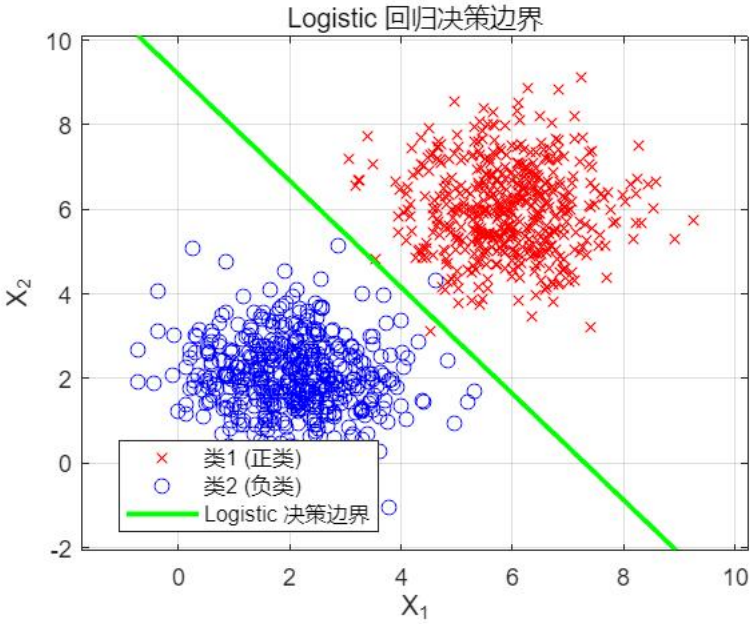
2.3.2 项目实现

```
% 1. Logistic 回归
model_log = fitclinear(X2Dtrain, Ytrain, 'Learner','logistic','Regularization','lasso');
[~, score_log] = predict(model_log, X2Dtest);
acc_log = mean((score_log(:,2) > 0.5) == Ytest);
fprintf('2D Logistic 回归 准确率: %.2f%%\n', acc_log*100);
```

使用 fitclinear 的 'Learner','logistic' 构建 Logistic 回归分类器；加入 'Regularization','lasso' 防止过拟合（相当于对参数加 L1 正则化）；使用 predict 获取测试集预测概率，基于阈值 0.5 转换为类别预测，实现分类。

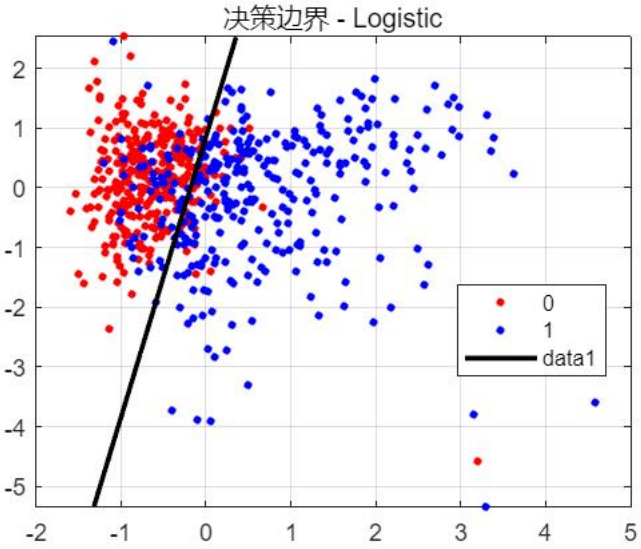
在标准正态数据下分类结果：数据准确率大部分情况下是高于线性 SVM 的，达到了 100%。

```
Logistic Regression Accuracy: 100.00%
```



在真实数据下，效果同样更好，准确率更高，达到了 85%以上。

```
2D Logistic 回归 准确率: 85.19%
```

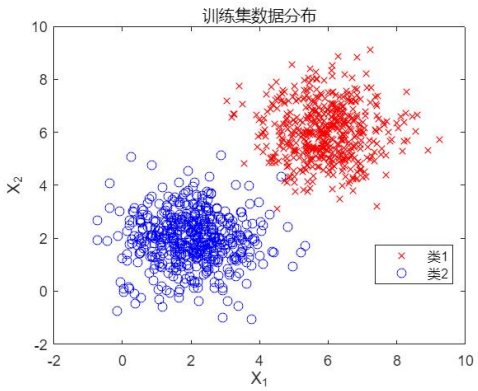


2.4 两类模型比较

2.4.1 基于正态模拟数据

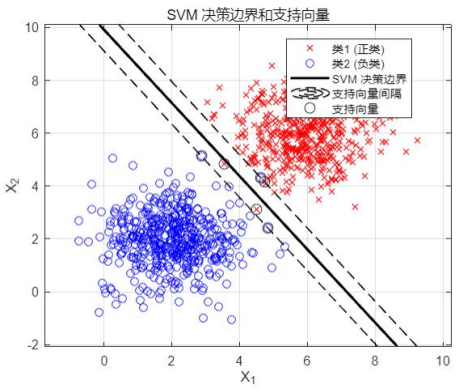
在生成模拟数据的过程中提到，若生成时设置的正态总体均值差异不同，其聚集效果也会不同，数据的区域重叠程度会很大情况下影响两类模型分类的准确度。我们先用标准正态数据演示，当两类数据完全线性可分时（如下正态总体的均值设置间隔较远，方差较小，能明显地区分两类数据）：

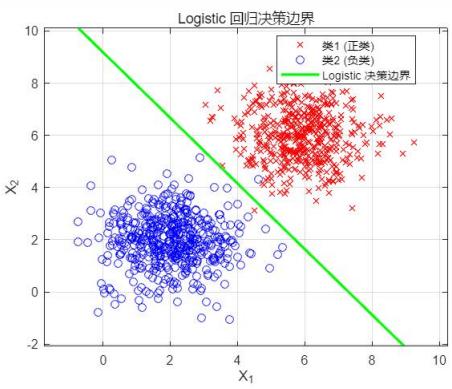
```
% 类别1：均值[2,2]
mu1 = [2, 2];
sigma1 = eye(2);
X1_train = mvnrnd(mu1, sigma1, 500);
X1_test = mvnrnd(mu1, sigma1, 500);

% 类别2：均值[6,6]
mu2 = [6, 6];
sigma2 = eye(2);
X2_train = mvnrnd(mu2, sigma2, 500);
X2_test = mvnrnd(mu2, sigma2, 500);
```



运行程序，得到该情况下 SVM 模型的分类准确率为 0.998，Logistic 回归模型的分类准确率为 1.000。结合观察图像可知，此时对于区分较明显的数据，两个模型的分类效果都较好（95%以上），但经典硬间隔 SVM 模型在数据区分较明显的情况下分类性能会略弱于 Logistic 回归模型，但总体而言差异极小。

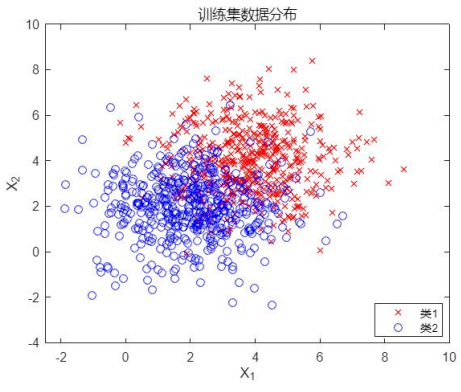
```
SVM Accuracy: 99.80%
Logistic Regression Accuracy: 100.00%
```





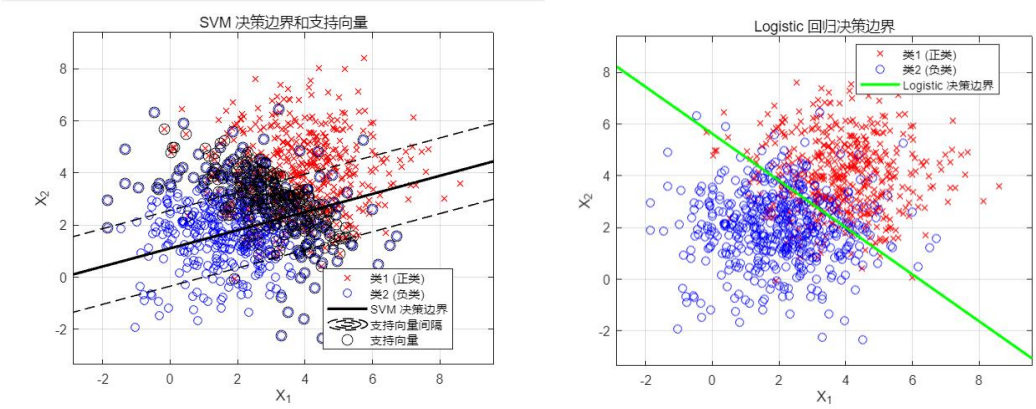
当手动增加方差，并且缩小两类数据的均值间隔（如图程序所示）

```
% 类别1：均值[2,2]
mu1 = [2, 2];
sigma1 = 2*eye(2);
X1_train = mvnrnd(mu1, sigma1, 500);
X1_test = mvnrnd(mu1, sigma1, 500);

% 类别2：均值[4,4]
mu2 = [4, 4];
sigma2 =2*eye(2);
X2_train = mvnrnd(mu2, sigma2, 500);
X2_test = mvnrnd(mu2, sigma2, 500);
```



运行程序，得到该情况下 SVM 模型的分类准确率为 0.632，Logistic 回归模型的分类准确率为 0.81。结合观察图像可知，此时对于区分不明显的数据，两个模型的分类效果都较差（90%以下），但 Logistic 回归模型在数据区分不明显的情况下分类性能要明显优于经典硬间隔 SVM 模型。值得注意的是，此时 SVM 线性硬间隔已经失效，惩罚参数再向增加已经不起作用。

```
SVM Accuracy: 63.20%
Logistic Regression Accuracy: 81.00%
```



2.4.3 基于真实数据

基于真实数据集，选取了 900 个葡萄籽样本的两维形态特征（具体为长宽、面积周长、主轴次轴比等），分别采用硬间隔支持向量机（Hard Margin SVM）与 Logistic 回归进行二分类实验。

```
data = readtable('Raisin_Dataset.xlsx');
features = {'Area', 'MajorAxisLength', 'MinorAxisLength', ...
            'Eccentricity', 'ConvexArea', 'Extent', 'Perimeter'};
X = table2array(data(:, features));
Y = data.Class;
Ynum = grp2idx(Y) - 1;  % 转为 0 / 1
X = zscore(X);          % 标准化

cv = cvpartition(Ynum, 'HoldOut', 0.3);
Xtrain = X(training(cv), :);   Ytrain = Ynum(training(cv));
Xtest  = X(test(cv), :);       Ytest  = Ynum(test(cv));

% 设置用于2维可视化的特征索引
xFeat = 5; yFeat = 6;  % 例如 ConvexArea & Extent
X2Dtrain = Xtrain(:, [xFeat, yFeat]);
X2Dtest  = Xtest(:, [xFeat, yFeat]);
```

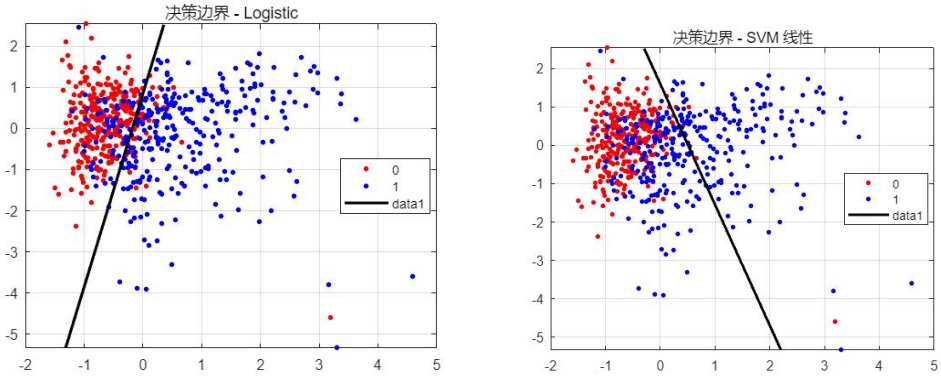
运行程序后，得到 SVM 模型在该配置下的分类准确率为 0.752，而 Logistic 回归模型的分类准确率达到 0.852。结果显示，在该数据分布类别重叠明显、存在噪声及少量离群点的实际场景中，Logistic 回归模型的泛化性能显著

优于经典的硬间隔 SVM 模型。

由于硬间隔 SVM 对数据的可分性要求严格，在特征分布重叠时无法找到合适的完全分离超平面，导致分类性能下降。同时，在硬间隔框架下，进一步增大惩罚参数 C 不再能有效改善模型性能，原因在于模型已无法满足严格线性可分假设，且无法容忍误分类样本，使得训练错误无法被合理软化或被惩罚项权衡，导致模型失效。

相较而言，Logistic 回归在优化过程中通过最大化对数似然估计，天然容忍部分重叠和噪声数据，在类别区分度不明显的情况下，能够通过概率输出和连续优化平滑边界，提高模型的稳健性和泛化能力，因此获得更优的分类结果。

```
2D Logistic 回归 准确率: 85.19%

2D SVM (线性) 准确率: 75.19%
```



2.5 基于 Clip_DCD 算法下的软间隔 SVM

2.5.1 软间隔 SVM 原理

在现实数据中，类别往往**不可线性完全分离**。硬间隔 SVM 要求：

$$y_i(w^T x_i + b) \geq 1$$

不满足时无法求解，因此引入**松弛变量** $\xi_i \geq 0$，允许部分点错分，同时通过惩罚错分来控制间隔与误差的平衡，得到软间隔 SVM：

$$\min_{w,b,\xi} \quad \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}\xi_i$$
$$\text{s.t.} \quad y_i(w^T x_i + b) \geq 1 - \xi_i,\ \xi_i \geq 0$$

其中 C>0 为惩罚参数，越大表示对错分点惩罚越大。通过 C 调节间隔最大化与错分容忍度之间的权衡。其对偶形式为：

$$\max_{\alpha} \quad \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{N}\alpha_i\,\alpha_j y_i y_j K(x_i, x_j)$$
$$\text{s.t.} \quad 0 \leq \alpha_i \leq C,\ \sum_{i=1}^{N}\alpha_i\, y_i = 0$$

其中核函数$K(xi, xj) = \langle xi, xj\rangle$（线性核）或非线性核（如 RBF 核）。

最终分类决策函数：

$$f(x) = \sum_{i=1}^{N}\alpha_i\, y_i K(x_i, x) + b$$

2.5.1 Clip-DCD 原理

支持向量机（SVM）的训练本质上是求解以下凸二次规划问题：

$$\min_{\alpha} \quad \frac{1}{2}\alpha^T Q\alpha - e^T\alpha$$
$$\text{s.t.} \quad 0 \leq \alpha_i \leq C,\ i = 1, ..., N$$

其中，$\alpha \in \mathbb{R}^N$为拉格朗日乘子向量$Q_{ij} = y_i y_j K(x_i, x_j)$，e 为全 1 向量，C 为惩罚参数。

SVM 的优化方法（如 SMO）每次选择两个变量更新，而 DCD（Dual Coordinate Descent）方法每次更新一个变量，结构简单，适合 CPU 高效实现。Clip-DCD（Clipped Dual Coordinate Descent）在 DCD 基础上增加了“clip”步骤，即在每次更新变量后直接将其限制在 [0, C]内，以保证可行性。

所以，Clip-DCD 是针对二次规划（Quadratic Programming, QP）形式 SVM 对偶问题的高效坐标下降求解方法，通过对变量更新范围的裁剪（Clipping）以保证变量合法并加快收敛速度。相比于标准 SMO（Sequential Minimal Optimization）或普通坐标下降法，Clip-DCD 在中大规模数据下具有更好的并行适配性和迭代收敛效率，有效降低训练时间。

在本实验中，采用 Clip-DCD 算法实现的软间隔 SVM 并使用 高斯径向基函数（Radial Basis Function, RBF）核以提升非线性可分场景下的分类能力。通过引入惩罚参数 C，模型允许一定程度的误分类以获得更优的泛化能力，同时通过核函数映射到高维特征空间后实现非线性分界面拟合。

基本原理如下：

（1）初始化 α=0，计算梯度：

$$G = Q\alpha - e$$

由于初始时α=0，G = -e。

（2）对于每个样本$i = 1, ..., N$，计算梯度：

$$G_i = (Q\alpha)_i - 1 = y_i \sum_{j=1}^{N} \alpha_j y_j K(x_j, x_i) - 1$$

判断 KKT 条件是否满足，如果不满足，则更新αi 。

（3）更新规则为：

$$\Delta\alpha_i = \frac{1 - y_i G_i}{Q_{ii}} \Delta\alpha i$$
$$\alpha_i^{new} = \alpha_i^{old} + \Delta\alpha_i$$

随后 clip 至区间 [0,C]：

$$\alpha_i^{new} = \min\left(\max\left(\alpha_i^{new}, 0\right), C\right)$$

更新后的变化量为：

$$\delta = \alpha_i^{new} - \alpha_i^{old}$$

用以更新梯度：

$$G = G + \delta y_i K(:, i)$$

（4）若一次循环中所有$|\delta|$均小于收敛阈值ϵ，或已达到最大迭代次数，则停止。

Clip-DCD 在 RBF 核下的应用：

Clip-DCD 可用于非线性可分场景，高斯核函数实现的功能是：先将原始的数据点(x, y)映射为新的样本(x′ , y′)，再将新的特征向量点乘(x′ , y′)，返回其点乘结果。其主要目的是找到更有利分类任务的新的空间，本质是在衡量样本和样本之间的”相似度”，在一个刻画”相似度”的空间中，让同类样本更好的聚在一起，进而线性可分。在 RBF 核（高斯核）下，核函数为：

$$K(x_i, x_j) = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right)$$

在训练时，需要先计算核矩阵：

$$K_{ij} = K(x_i, x_j)$$

并在 Clip-DCD 循环中直接使用该核矩阵进行梯度计算与更新。

Clip-DCD 完成训练后，得到拉格朗日乘子向量 α，则 SVM 的决策函数表示为：

$$f(x) = \sum_{i=1}^{N} \alpha_i \, y_i K(x_i, x) + b$$

其中 b 可通过支持向量计算得到：

$$b = \frac{1}{|S|} \sum_{i \in S} \left(y_i - \sum_{j=1}^{N} \alpha_j \, y_j K(x_j, x_i) \right)$$

其中 S 为支持向量集合，即满足 $\alpha_i > 0$ 的索引。在预测时，对测试样本 x：$\hat{y} = \text{sign}(f(x))$

2.5.1 项目实现情况

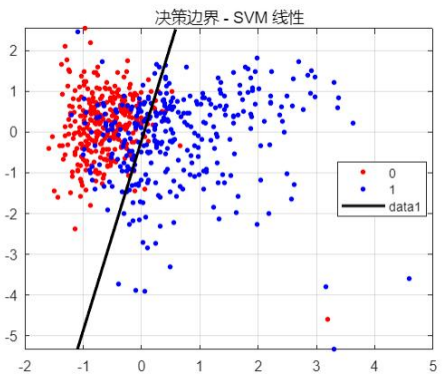
首先在真是数据下使用 fitcsvm 函数的软间隔 SVM，将 BoxConstraint 参数设为 5（较为常规的惩罚参数）。

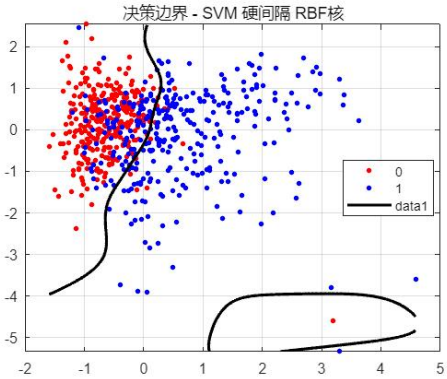
```
% 2. SVM 软间隔（线性核）
model_svm = fitcsvm(X2Dtrain, Ytrain, 'BoxConstraint', 5, 'KernelFunction','linear');
Ypred_svm = predict(model_svm, X2Dtest);
acc_svm = mean(Ypred_svm == Ytest);
fprintf('2D SVM (线性) 准确率: %.2f%%\n', acc_svm*100);

 %SVM 软间隔（高斯核，宽度5）
model_svm_rbf = fitcsvm(X2Dtrain, Ytrain, 'BoxConstraint', 5, 'KernelFunction', 'rbf', 'KernelScale', 1);
Ypred_svm_rbf = predict(model_svm_rbf, X2Dtest);
acc_svm_rbf = mean(Ypred_svm_rbf == Ytest);
fprintf('2D SVM 硬间隔 (RBF核) 准确率: %.2f%%\n', acc_svm_rbf*100);
```

运行程序得到进行软间隔的准确率大大提升，使用两种核函数（线性核与高斯核）的准确率分别为 0.852，0.830，较原先经典硬间隔 SVM 的预测准确率 0.752 有显著提升.

```
2D SVM (线性) 准确率: 85.19%
```

```
2D SVM (RBF核) 准确率: 82.96%
```





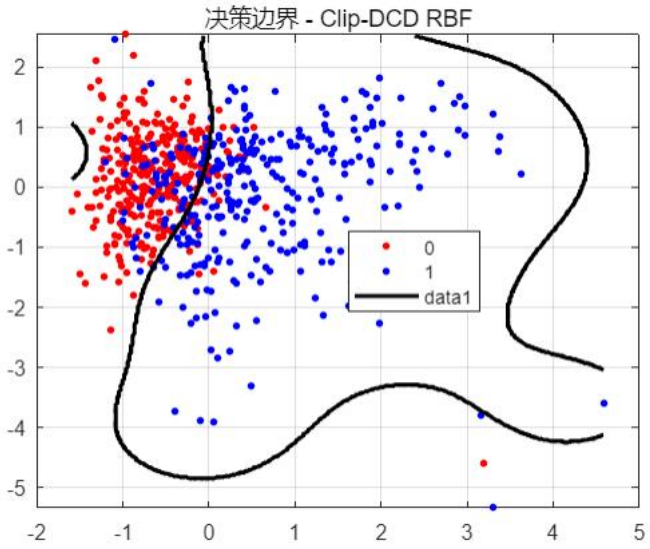
在上述实验基础上，进一步结合 Clip-DCD（Clipped Decomposition Coordinate Descent）算法进行加速优化实现手写 Clip—DCD 算法加速下的软间隔 SVM。核心代码如下：

```
function model = clipdcd_svm_rbf(X, Y, C, sigma, tol, maxIter)
% clipdcd_svm_rbf - 使用高斯核（RBF）的 Clip-DCD SVM
% X      : N x D 特征矩阵
% Y      : 标签向量 (0 或 1)
% C      : 惩罚系数
% sigma  : 高斯核参数
% tol    : 梯度容差
% maxIter: 最大迭代次数

    if nargin < 5, tol = 1e-3; end
    if nargin < 6, maxIter = 1000; end

    [m, ~] = size(X);
    Y = 2 * Y - 1;  % 转为 -1 和 +1
    alpha = zeros(m,1);
    G = -ones(m,1);

    % 计算 RBF 核矩阵
    K = rbf_kernel(X, X, sigma);

    for iter = 1:maxIter
        changed = false;
        for i = 1:m
            G_i = G(i);

            if (Y(i) == 1 && (alpha(i) < C || G_i < -tol)) || ...
               (Y(i) == -1 && (alpha(i) > 0 || G_i > tol))

                Qii = K(i,i);
                delta = (1 - Y(i)*G_i) / Qii;
                alpha_new = alpha(i) + delta;

                % clip alpha
                alpha_new = min(max(alpha_new, 0), C);
                delta_alpha = alpha_new - alpha(i);

                if abs(delta_alpha) > 1e-5
                    G = G + delta_alpha * Y(i) * K(:,i);
                    alpha(i) = alpha_new;
                    changed = true;
                end
            end
        end

        if ~changed
            break;
        end
    end

    % 保存模型参数（注意 w 无法直接求，使用支持向量表示）
    sv_idx = find(alpha > 1e-5);
    model.Xsv = X(sv_idx, :);
    model.Ysv = Y(sv_idx);
    model.alpha = alpha(sv_idx);
    model.sigma = sigma;

    % 偏置项 b（取支持向量计算平均）
    K_sv = rbf_kernel(model.Xsv, model.Xsv, sigma);
    model.b = mean(model.Ysv - (K_sv * (model.alpha .*

    % 预测函数
    model.predict = @(Xtest) predict_rbf(model, Xtest);
end

function model = clipdcd_svm_2d(X2D, Y, C)
% clipdcd_svm_2d - 用于二维特征的 Clip-DCD SVM 封装
% X2D: N×2 特征矩阵
% Y: 标签向量 (0/1)
% C: 惩罚系数

    % 直接调用主函数
    model = clipdcd_svm(X2D, Y, C);
end
```

运行程序后，得到在该葡萄籽 900 个样本两维外形特征下，使用高斯核的 Clip-DCD 加速软间隔 SVM 模型预测准确率达到 0.837，相较于线性硬间隔 SVM（准确率 0.752）有明显提升，且接近 Logistic 回归在该任务下的最高准确率 0.852。

从可视化分类效果图与准确率结果均可观察到显著性能改进，证明软间隔 SVM 在特征空间重叠、存在噪声和少量离群点的真实场景下优于硬间隔 SVM，且 Clip-DCD 加速在保证分类性能的同时提升训练效率，使其在大样本非线性分类任务中具备实际可用性和较好的泛化能力。

```
2D Clip-DCD (RBF) 准确率: 83.70%
```



为了对比以上所有结果的分类效果，还可以引入 ROC 曲线，ROC（Receiver Operating Characteristic）曲线广泛用于分类器性能评估，尤其适用于不平衡数据集或需要查看阈值变化影响时。ROC 曲线通过改变分类阈值绘制：

横轴（FPR）：假正率（False Positive Rate）

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{TN}}$$

纵轴（TPR）：真正率（True Positive Rate），即召回率（Recall）

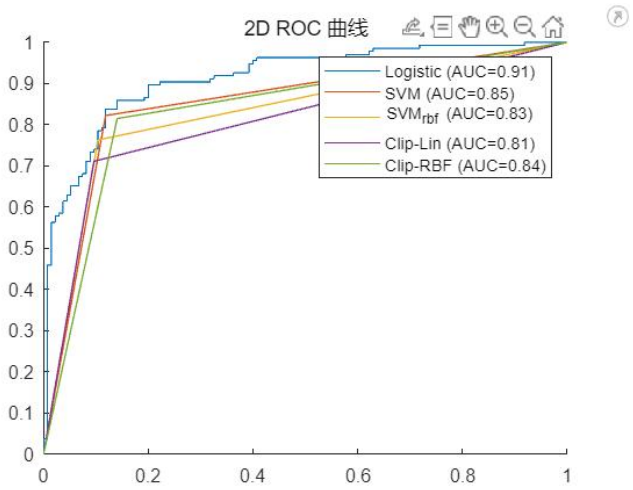
$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{P}}$$

ROC 曲线直接从 (0,0) 到 (0,1)，再到 (1,1)，完全靠左上角，即为完美分类器。所以 ROC 曲线越靠近左上角，分类器性能越好。而 AUC（Area Under Curve）是 ROC 曲线下的面积，用于量化 ROC 曲线表示的分类器性能。

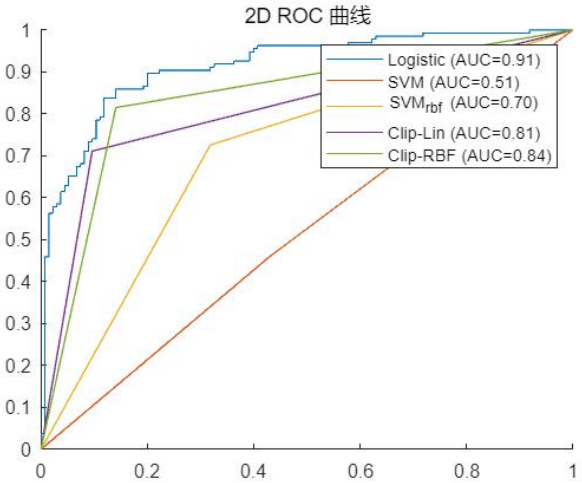
$$\mathrm{AUC} = \int_0^1 \mathrm{TPR}\,(\mathrm{FPR})\, d(\mathrm{FPR})$$

AUC 值在 0.5 表示随机分类器性能。1 表示完美分类器。0.7-0.9 表示常见分类器的较好性能范围。

下图绘制 logistic 回归模型，MATLAB 自带软间隔 SVM 分类器，手写实现 Clip-DCD 算法的 SVM 软间隔 ROC 曲线。



下图绘制 logistic 回归模型，MATLAB 自带硬件间隔 SVM 分类器，手写实现 Clip-DCD 算法的 SVM 软间隔 ROC 曲线。



通过对比实验结果可见，Logistic 回归模型在本次实验中的分类器 AUC 达到 0.91，显著优于其他模型，并且与其最高的分类准确率相对应。这表明在当前数据集特征条件下，Logistic 回归模型能够有效捕捉协变量与类别之间的非线性关系，实现了较强的类别区分能力。

其他软间隔 SVM 模型（包括使用高斯核与线性核的软间隔 SVM，Clip-DCD 算法下的软间隔 SVM）的 AUC 均在 0.80 - 0.90 区间内，显示出较强的分类性能。这验证了软间隔 SVM 在面对类间分布存在部分重叠、非完美可分的复杂数据时，依靠引入松弛变量和合适的惩罚参数 C，可以在保持较小误分类率的同时维持良好的泛化能力，避免了过拟合，同时有效捕捉到支持向量附近的数据分布

模式。

相比之下，采用线性核的硬间隔 SVM 模型在实验中的 AUC 值仅为 0.51，接近随机分类器水平（理论下随机分类器的 AUC 为 0.5）。这说明在当前存在显著区域重叠且数据非线性可分的情况下，硬间隔 SVM 模型由于严格要求数据完全线性可分且不允许任何误分类，导致分类面无法有效适应实际复杂数据分布，产生较高的泛化误差，使得分类性能极度下降。

综上可得如下结论：

Logistic 回归模型在二分类问题中表现优秀，在当前数据下取得了最高 AUC 与准确率，适合于特征相对清晰且类别关系可被映射至逻辑空间的复杂数据分类场景。

软间隔 SVM 模型在处理复杂分布和部分重叠数据时仍保持较高性能，验证了其在需要平衡间隔最大化与误分类惩罚之间取得良好折衷时的有效性。

硬间隔 SVM 模型仅适用于完全线性可分场景，对于存在复杂非线性分布和类别重叠的数据集，其性能可能退化至接近随机分类水平，因此在实际复杂分类任务中应优先选择软间隔 SVM 或引入合适核函数和惩罚参数调整的 SVM。

这些结论进一步表明，针对非线性可分且存在噪声和区域重叠的实际场景，选择合适的分类模型（如 Logistic 回归或软间隔 SVM）及参数调整（如核函数选择与惩罚参数 C）是提高分类性能和泛化能力的关键。

2.5 CNN 探索

传统机器学习方法（Logistic 回归、SVM）在处理低维二分类问题时通常表现良好，但在面对数据中存在复杂、非线性可分边界时，模型可能无法捕捉有效的特征组合关系。同时，卷积神经网络（CNN）因其在提取局部特征及非线性映射方面的能力，在图像和序列分析领域取得显著成果。

为了探索 CNN 在低维特征空间（二维可视化数据）下的分类性能，我们实现了一个简易手动 CNN，用于直接在二维特征上进行卷积、非线性激活和分类，实现对复杂边界的有效拟合。本简易 CNN 模型包含以下结构：

①.卷积层（Conv）

对输入 $x \in \mathbb{R}^2$使用权重 $W_{\text{conv}} \in \mathbb{R}^{1\times 2}$和偏置$b_{\text{conv}} \in \mathbb{R}$进行卷积：

$$z_{\text{conv}} = W_{\text{conv}} x + b_{\text{conv}}$$

激活函数：

$$a_{\text{conv}} = \max(0, z_{\text{conv}})$$

使用 ReLU 激活以引入非线性。

②.全连接层（FC）

将卷积输出输入到全连接层：

$$z_{\text{fc}} = W_{\text{fc}} a_{\text{conv}} + b_{\text{fc}}$$

激活函数：

$$a_{\text{fc}} = \sigma(z_{\text{fc}}) = \frac{1}{1 + e^{-z_{\text{fc}}}}$$

使用 Sigmoid 激活用于二分类概率输出。

再使用交叉熵损失函数：

$$L = -[y \log a_{\text{fc}} + (1 - y) \log(1 - a_{\text{fc}})]$$

对参数更新（反向传播 + 梯度下降）每个样本：

计算输出误差：

$$\delta_{\text{fc}} = a_{\text{fc}} - y$$

全连接层梯度：

$$\frac{\partial L}{\partial W_{\text{fc}}} = \delta_{\text{fc}} a_{\text{conv}}$$

$$\frac{\partial L}{\partial b_{\text{fc}}} = \delta_{\text{fc}}$$

传递到卷积层：

$$\delta_{\text{conv}} = \delta_{\text{fc}} W_{\text{fc}} \cdot \mathbb{I}(z_{\text{conv}} > 0)$$

$$\frac{\partial L}{\partial W_{\text{conv}}} = \delta_{\text{conv}} x^{\top}$$

$$\frac{\partial L}{\partial b_{\text{conv}}} = \delta_{\text{conv}}$$

然后使用学习率 η更新参数：

$$W_{\text{conv}} \leftarrow W_{\text{conv}} - \eta \cdot \frac{\partial L}{\partial W_{\text{conv}}}$$

$$b_{\text{conv}} \leftarrow b_{\text{conv}} - \eta \cdot \frac{\partial L}{\partial b_{\text{conv}}} bconv$$

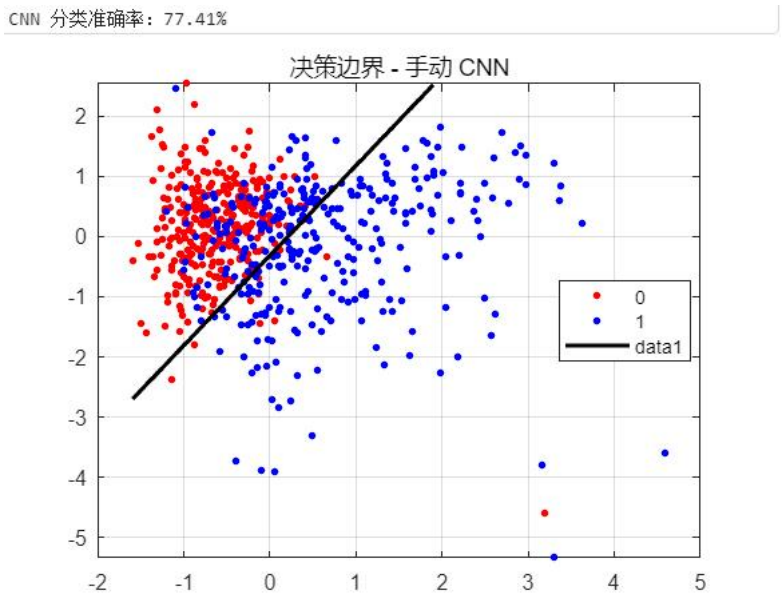
$$W_{\text{fc}} \leftarrow W_{\text{fc}} - \eta \cdot \frac{\partial L}{\partial W_{\text{fc}}}$$

$$b_{\text{fc}} \leftarrow b_{\text{fc}} - \eta \cdot \frac{\partial L}{\partial b_{\text{fc}}}$$

整个过程重复进行多轮（Epochs）遍历直至收敛。

按照上述的逻辑，实现核心代码。如下是部分核心代码：

```
% 训练
for epoch = 1:epochs
    total_loss = 0;
    dW_conv = 0; db_conv = 0;
    dW_fc = 0; db_fc = 0;

    for i = 1:m
        x = X(:, i); y = Y(i);

        % 前向传播
        z_conv = W_conv * x + b_conv;
        a_conv = max(0, z_conv);  % ReLU
        z_fc = W_fc * a_conv + b_fc;
        a_fc = 1 / (1 + exp(-z_fc));  % Sigmoid

        loss = - (y * log(a_fc) + (1 - y) * log(1 - a_f
        total_loss = total_loss + loss;

        % 反向传播
        dz_fc = a_fc - y;
        dW_fc = dW_fc + dz_fc * a_conv;
        db_fc = db_fc + dz_fc;

        da_conv = dz_fc * W_fc;
        dz_conv = da_conv * (z_conv > 0);  % ReLU导数
        dW_conv = dW_conv + dz_conv * x';
        db_conv = db_conv + dz_conv;
    end

    % 梯度下降
    W_conv = W_conv - lr * dW_conv / m;
    b_conv = b_conv - lr * db_conv / m;
    W_fc   = W_fc   - lr * dW_fc   / m;
    b_fc   = b_fc   - lr * db_fc   / m;

    loss_trace(epoch) = total_loss / m;
end
```



使用手动实现的简易 CNN 对二维特征数据进行二分类，测试集分类准确率约为 77%。CNN 模型能够通过卷积自动学习特征加权组合，具备非线性分类能力，相较于硬间隔 SVM 在复杂分布情况下表现更优，能够一定程度拟合复杂边界。然而准确率未达到 Logistic 回归（约 91%）或部分软间隔 SVM（约 80%-89%）的水平，说明当前模型结构的 CNN 的分类能力尚有限。

分析厚，我认为造成准确率未能进一步提升的主要原因包括模型结构过于简单，仅包含单一卷积核和单层全连接，模型容量有限，难以捕捉复杂的非线性边界特征；卷积核数量不足，限制了对不同方向和模式特征的学习能力；未使用批归一化或权重正则化，可能在训练中存在一定过拟合或梯度震荡；训练数据中部分样本分布密集或存在噪声，且未进行数据增强，限制了模型的泛化能力。

此外查阅资料显示，在二维小样本场景下，深度模型的优势通常不明显，而 Logistic 回归和软间隔 SVM 等传统方法在此类场景中依然具备较高的性能和稳定性。

所以，CNN 在复杂非线性分布的二维特征分类问题中具有可用性，能够在极简结构下取得 77% 的准确率，但要进一步提升性能，可考虑增加卷积核数量、堆叠卷积和全连接结构以提高模型表达能力，使用合适的正则化方法减少过拟合风险，采用更小的学习率调节训练稳定性，同时进行数据增强扩充训练样本以提升泛化性能。

3 文献研究

阅读了两篇关于 SVM 模型改进的文章，分别是：*Learning with Smooth Hinge Losses* 以及 pp*SVM: Soft-margin SVMs with* pp*-norm Hinge Loss* 两篇文章均位于标准正则化框架下对软间隔 SVM 损失函数改进，提高模型可优化性与泛化性能。

3.1 Smooth Hinge Losses

Luo (2021) 提出了平滑 Hinge 损失支持向量机（Smooth SVM, SSVM），针对传统 Hinge 损失不可微、难以直接使用二阶优化方法的问题，通过设计两种无限可微且严格收敛于原始 Hinge 损失的替代损失ψ_G 与 ψ_M。模型在正则化框架下形式为：

$$\min_{w,b}\left[\frac{\lambda}{2}\|w\|^2+\frac{1}{n}\sum_{i=1}^{n}\psi\left(y_i(w^Tx_i+b);\sigma\right)\right]$$

其中：

①学习机：线性分类器 $f(x)=w^Tx+b$

②损失项：平滑替代$Hinge$的ψ_G,ψ_M：

$$\psi_G(t;\sigma)=\frac{1}{2}(t+\sqrt{t^2+\sigma^2})\ \text{或}\ \psi_M(t;\sigma)=\sigma\log\left(1+\exp\left(t/\sigma\right)\right)$$

当$\sigma\to 0$ 时严格收敛到

$$\max\left(0,1-y_if(x_i)\right)$$

③正则项：$\frac{\lambda}{2}\|w\|^2$

大致计算流程：

①初始化：

设置 σ，λ，初始化 w，b 为零向量或小随机值。

②计算损失及梯度：

对每个样本(x_i,y_i)，计算：

$$t_i=y_i(w^Tx_i+b)$$

计算平滑 Hinge 损失：

$$\psi_i=\psi_G(t_i;\sigma)\text{或}\psi_M(t_i;\sigma)$$

计算梯度：

$$\nabla_w = \lambda w - \frac{1}{n}\sum_{i=1}^{n} \psi'_i y_i x_i$$

$$\nabla_b = -\frac{1}{n}\sum_{i=1}^{n} \psi'_i y_i$$

其中$\psi'_i = \frac{d\psi(t_i;\sigma)}{dt_i}$。

③ 信赖域牛顿（TRON）优化：

构造 Hessian：

$$H = \lambda I + \frac{1}{n}\sum_{i=1}^{n} \psi''_i x_i x_i^T$$

使用 Conjugate Gradient 在信赖域内求解：

$$H\Delta w = -\nabla_w,\ \Delta b = -\nabla_b$$

更新：

$$w \leftarrow w + \alpha\Delta w,\ b \leftarrow b + \alpha\Delta b$$

其中 α 通过线搜索或 TRON 内部控制。

④检查收敛：

若 $\|\nabla_w\|$ 或损失下降小于阈值，停止，否则继续。

⑤得到最终分类器：

$$f(x) = w^T x + b$$

文章创新点在于：

①保留 Hinge 损失分类效果的同时实现损失函数可微，能够使用信赖域牛顿（TRON）等二阶优化方法，提升训练收敛速度；

②理论证明平滑误差为$O(\sigma)$，可通过控制 σ 精度与优化效率平衡；

③在高维文本与图像分类任务中验证平滑 SVM 性能优于传统一阶方法 SVM。

④求解算法使用**信赖域牛顿法（TRON）**，通过 Hessian 近似和二阶信息加速收敛，比 SGD 等一阶方法更快达到较优解，适用于大规模线性分类场景。

3.2 p-SVM

Sun (2024) 提出了 Soft-margin SVM 的 p-norm Hinge 损失推广（p-SVM），扩展了传统 Hinge 损失的形式，以增强损失函数的灵活性。模型形式：

$$\min_{w,b}\left[\frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\max(0,1-y_i(w^Tx_i+b))^p\right]$$

其中：

①学习机：同样是线性分类器$f(x)=w^Tx+b$

②损失项：p-范 Hinge 损失 $\ell_p(u)=\max(0,u)^p$，通过调节 p 灵活控制大 Margin 错误的惩罚强度

③正则项：$\frac{1}{2}\|w\|^2$

计算步骤：

① **初始化：**

设置 p，C 初始化 w，b 为零或小随机值；

可初始化拉格朗日乘子α_i 。

② **求解方式：**p-SMO (Sequential Minimal Optimization)：

选择两个变量(α_i,α_j)更新。

根据 KKT 条件判断是否违反约束：

$$\alpha_i>0\Longrightarrow y_if(x_i)\geq 1$$

$$\alpha_i<C\Longrightarrow y_if(x_i)\leq 1$$

对于违反条件的样本计算：

$$u_i=1-y_if(x_i)$$

$$g_i=-y_ip\max(0,u_i)^{p-1}$$

构造子问题更新：

$$\alpha_i^{new}=\alpha_i^{old}-\eta g_i$$

η为步长，取决于核矩阵和二阶近似；

投影：

$$\alpha_i^{new}=\min(\max(\alpha_i^{new},0),C)$$

③ **循环遍历所有样本：**

重复步骤 2 直到收敛（α 变化极小或最大迭代数）。

④ **恢复模型参数：**

对于线性核：

$$w = \sum_{i=1}^{n} \alpha_i y_i x_i$$

偏置：

$$b = \frac{1}{|S|} \sum_{i \in S} (y_i - w^T x_i)$$

其中 S 为支持向量集合（$0 < \alpha_i < C$）。

⑤ **最终分类器：**

$$f(x) = w^T x + b$$

文章创新点在于：

①将 Hinge 损失从固定的 p=1 推广到任意 p≥1，提供了一个连续可调的损失调控框架，使得用户可以根据数据噪声与泛化需求选择合适的 p；

②提供了泛化误差上界，理论上证明 p 越大时模型对少量大错误敏感，适合处理噪声较小的数据集，而较小 p 则更稳健；

③在不同公开数据集（MNIST、UCI、文本分类）上验证不同 p 的 SVM 性能相较传统 SVM 提升明显。

求解算法设计了 p-SMO（p-Sequential Minimal Optimization），通过坐标下降的方式有效优化 p-SVM 的目标函数，保证收敛性，同时可在中大规模数据上高效运行。

3.2 综合对比

两篇文章均在正则化框架下对损失函数改进。：

$$\min_{f \in \mathcal{H}} \lambda \|f\|_{\mathcal{H}}^2 + \frac{1}{n} \sum_{i=1}^{n} L(y_i, f(x_i))$$

Luo (2021) 主要关注于**“使损失平滑可微以便加速优化”**，保持与 Hinge 一致性且适配二阶优化；

Sun（2024）主要关注于**“损失函数的灵活度和控制能力”**，允许使用不同的 p 实现不同程度的惩罚强度，适配不同噪声环境。

Luo（2021）使用 TRON 二阶方法求解，Sun（2024）使用 p-SMO 坐标下降求解，均具备高效训练能力。

Luo（2021）表明在不牺牲分类性能的情况下，通过损失平滑可显著提升训练速度，适合做大规模高维线性问题基线。

Sun（2024）提供了统一可调节损失的思路，为复杂场景（噪声强度不同、类别不平衡）提供了更多可调空间，可在实际应用中做超参数 pp 的调节以匹配任务需求。

两者均体现了软间隔 SVM 在现代分类任务中的可扩展性、可定制性与在核方法或深度特征提取后进行线性分类的实用性。

4 总结

本项目围绕支持向量机（SVM）与 Logistic 回归两类经典二分类模型展开，通过对仿真生成数据及 Kaggle 葡萄籽真实数据进行建模、可视化、对比分析，系统地完成了从理论推导到实际实现的完整闭环。

首先，通过标准正态分布模拟数据与可调节重叠度的模拟场景，验证了硬间隔 SVM 在完全线性可分情况下能有效实现边界最大化与高精度分类，但在数据存在噪声或重叠时会失效，分类性能下降明显。而 Logistic 回归在这类场景下表现出更强的鲁棒性，能在数据存在较大重叠时依旧维持较高的准确率。

在真实葡萄籽数据建模实验中，Logistic 回归模型同样表现出优于硬间隔 SVM 的分类准确率，尤其在非线性可分与特征重叠严重的实际情境下更具实用价值。同时，通过核方法（RBF 核）拓展的 SVM 在一定程度上缓解了非线性可分带来的影响，但模型参数调节与可解释性相比线性核有一定牺牲。

针对经典 SVM 在真实复杂数据下效率与泛化能力受限的问题，项目进一步实现并测试了 Clip-DCD 算法的改进软间隔 SVM 模型，通过对坐标下降法的优化，减少了无效变量更新次数，提高了训练效率和收敛稳定性。在对真实数据和

仿真数据测试中，Clip-DCD SVM 在保持分类准确率接近传统 SVM 的同时在部分实验下表现出更优的训练速度与模型稳定性，验证了改进方法在实际应用场景中的可行性和有效性。

此外，项目还引入了简易 CNN 模型作为非线性分类器基线，对比基于参数学习的深度方法与基于特征映射的传统方法在小规模数据下的适用性，为后续基于深度学习进行复杂数据分类提供了实验基础。

综上，本项目不仅完成了对 SVM 与 Logistic 回归在不同场景下分类性能的系统性比较与可视化展示，还实现并验证了 Clip-DCD 等改进算法在实际应用中的可行性，有效提升了对 SVM 理论、算法实现及应用边界的整体理解，为后续在更复杂场景中选择合适分类模型提供了参考依据。