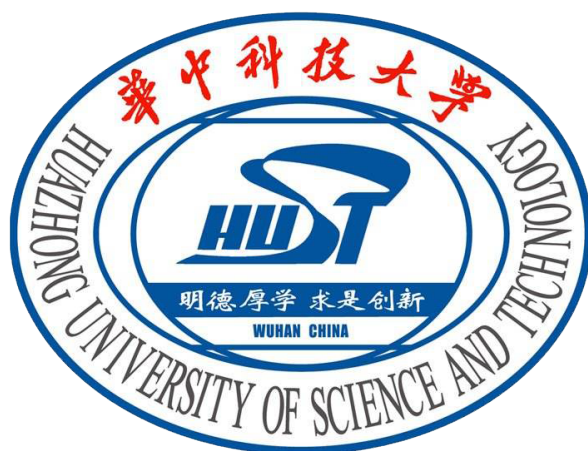


# 华中科技大学计算机科学与技术学院

## 《机器学习》 结课报告



专	业	<u>数据科学与大数据技术</u>
班	级	<u>大数据 2201</u>
学	号	
姓	名	
成	绩	_____
指导教师		<u>张腾</u>
时	间	<u>2024 年 5 月 21 日</u>

# 目录

<b>1</b>	<b>实验要求</b>	<b>1</b>
<b>2</b>	<b>算法设计与实现</b>	<b>2</b>
2.1	KNN 算法 . . . . .	2
2.2	MLR 算法 . . . . .	3
2.3	MSVM 算法 . . . . .	4
<b>3</b>	<b>实验环境与平台</b>	<b>6</b>
3.1	硬件环境 . . . . .	6
3.2	测试环境 . . . . .	6
3.3	依赖的 python 相关包 . . . . .	6
<b>4</b>	<b>结果与分析</b>	<b>7</b>
4.1	算法测试与结果评估 . . . . .	7
4.2	结果分析 . . . . .	7
<b>5</b>	<b>个人体会</b>	<b>9</b>
	<b>参考文献</b>	<b>10</b>

## 1 实验要求

报告选择的题目是“数字识别”，本人按照任务包中 README.md 文件中的要求，分别采用了三种不同的机器学习模型对任务包中给定数据集进行训练和预测，并给出相应的模型评估。

“数字识别”任务包中给定的数据集来自 MNIST(Modified National Institute of Standards and Technology)，这是一个广泛用于机器学习和计算机视觉领域的基准数据集，由美国国家标准与技术研究院 (NIST) 的原始数据集修改而来。任务包已对 MNIST 数据集进行预处理并划分为 train 和 test 两个 csv 文件，需要说明的是它的一些关键特点：

- 图像大小：每个图像都是 28x28 像素的灰度图像。
- 类别：数据集共包含 10 个类别，分别对应数字 0 到 9。
- 训练集和测试集：数据集被分为两部分，train 文件中包含 42000 个训练样本，每个样本有一个长度为 784 的特征向量和一个类别标签；test 文件中包含 28000 个预测样本，每个样本仅包含一个特征向量，未给定相应标签，需要进行预测。本实验的 test 文件经过修改，添加了 label 列，数据摘自 kaggle 网站正确率较高的已提交结果（在 99.7% 以上），主要是方便本人在本地对比预测结果，从而得到相应的准确率，提前筛查出表现不佳的原因。

因此，手写数字识别是一个多类别分类问题，在本次任务中的目标是将图像对应的特征向量正确地分类为相应的数字。

在报告后面几部分，将给出本次实验采用的几种机器学习算法的设计思想及其实现过程，并根据本地对比结果与 kaggle 提交结果，对各个模型的性能进行评估，并分析模型存在的不足，力图分析一些模型表现不佳的原因和给出相应的优化方案。

## 2 算法设计与实现

在本实验中，一共采用了  $k$  近邻算法 (KNN)、多分类支持向量机算法 (Multiclassification Support Vector Machine, 下面简称 MSVM)、多元逻辑回归分析 (Multinomial Logistic Regression, 下面简称 MLR) 等多种机器学习算法对数据集进行建模和训练，并得到相应的预测结果。按照实验要求，KNN、MSVM、MLR 模型都是手动构建，没有调用已经封装好的相应机器学习库。下面是算法过程的数学描述。

### 2.1 KNN 算法

KNN 算法的具体过程如下：

1. 遍历训练数据集，计算预测样本与其他每一个样本点的距离，按照由近到远排序；本实验算法采用 Euclidean Distance（欧氏距离）。
2. 需要预先定义一个超参数参数  $k$  值， $k$  是一个大于或等于 1 的整数，表示纳入投票决策的样本数。本实验中选择  $k$  为 3。
3. 统计离预测样本最近的  $k$  个样本，统计各个分类数目，返回数量最多的类的标签作为预测值。

使用数学语言描述如下：

- 输入：训练数据集

$$T = (x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)$$

其中， $x_i \in X \subseteq \mathbb{R}^n$  为实例的特征向量（本实验中为长度为 784 的像素特征向量）， $y_i \in Y = \{c_1, c_2, \dots, c_k\}$  为实例的类别（本实验中为数字标签）， $i = 1, 2, \dots, N$ ；

- 输出：实例  $x$  所属的类  $y$

1. 根据给点的距离度量，在训练集  $T$  中找出与  $x$  最近邻的  $k$  个点，涵盖着  $k$  个点的领域，记为  $N_k(x)$ ；
2. 在  $N_k(x)$  中根据分类决策规则（如多数表决），决定  $x$  的类别  $y$ ：

$$y = \arg \max_{c_j} \sum_{x_i \in N_k(x)} I(y_i = c_j), \quad i = 1, 2, \dots, N$$

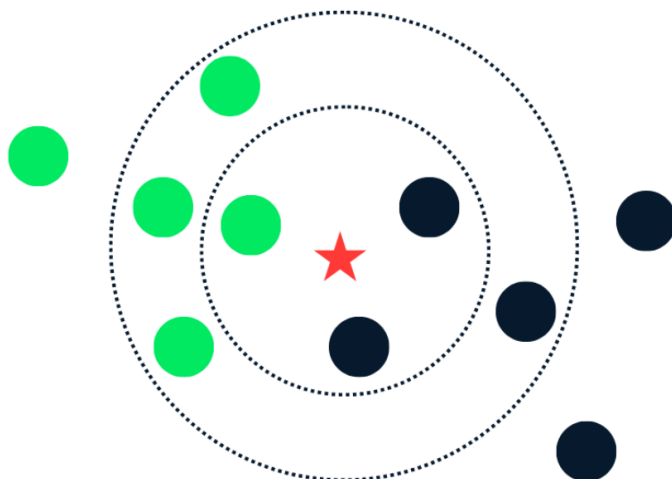


图 2.1: KNN 分类示例

## 2.2 MLR 算法

我们先简要描述一下二元逻辑回归模型。构建模型的过程如下：

1. 根据样本的特征确定属性矩阵，并为每个属性确定一个权重，生成一个权重矩阵  $w$ ，初始化为 0。同时，还要设置一个偏移矩阵  $b$ 。
2. 定义 Sigmoid 函数，将矩阵输出压缩到 0 与 1 之间作为样本属于正类的概率。

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

3. 使用交叉熵损失函数来衡量预测值与真实值之间的差异。对于单个样本，损失函数为：

$$L(y, \hat{y}) = -y \log(\hat{y}) - (1 - y) \log(1 - \hat{y})$$

其中， $y$  是真是标签（0 或 1）， $\hat{y}$  是预测概率。而对于整个数据集，总体损失是所有样本损失的平均值：

$$J(w, b) = \frac{1}{m} \sum_{i=1}^m L(y^{(i)}, \hat{y}^{(i)})$$

4. 用随机梯度下降法来最小化损失函数，通过对以上函数求导更新参数  $w$  和  $b$ ：

$$w_i = w_i - \alpha \frac{\partial J}{\partial w_i}$$

$$b = b - \alpha \frac{\partial J}{\partial b}$$

5. 重复上述步骤，直到损失函数的值收敛或达到预设的迭代次数。

有了 LR 算法，我们可以自然推广至 MLR 算法。使用数学语言描述如下：

- 输入：训练数据集

$$T = (x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)$$

其中,  $x_i \in X \subseteq \mathbb{R}^n$  为实例的特征向量 (本实验中为长度为 784 的像素特征向量),  $y_i \in Y = \{c_1, c_2, \dots, c_k\}$  为实例的类别 (本实验中为数字标签),  $i = 1, 2, \dots, N$ ;

- 输出：实例  $x$  所属的类  $y$

1. 对于每个类别  $c_j$ , 训练一个二元逻辑回归分类器, 其中将类别  $c_j$  标记为正类别, 而将其他所有类别标记为负类别;
2. 对于类别  $c_j$ , 的逻辑回归模型, 使用对数几率函数 (logit 函数) 来预测样本属于类别  $c_j$  的概率  $P(y = 1|x)$ , 即

$$P(y = 1|x) = \frac{1}{1 + e^{-\theta_j^T x}}$$

3. 对于其他所有类别  $c_l (l \neq j)$ , 预测概率为  $P(y = 0|x)$ :

$$P(y = 1|x) = 1 - P(y = 0|x)$$

4. 对于每一个样本, 最终的预测结果为概率最高的类别:

$$y = \arg \max_{c_j} P(y = 1|x)$$

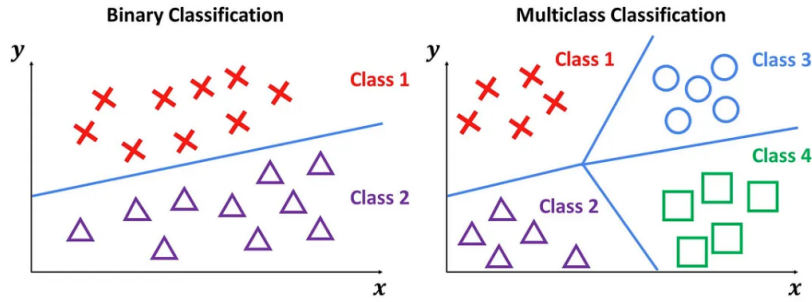


图 2.2: LR 与 MLR 分类示例

## 2.3 MSVM 算法

使用数学语言描述如下:

- 输入：训练数据集

$$T = (x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)$$

其中,  $x_i \in X \subseteq \mathbb{R}^n$  为实例的特征向量 (本实验中为长度为 784 的像素特征向量),  $y_i \in Y = \{c_1, c_2, \dots, c_k\}$  为实例的类别 (本实验中为数字标签),  $i = 1, 2, \dots, N$ ;

- 输出：实例  $x$  所属的类  $y$

1. 初始化权重矩阵  $w$  和偏置向量  $b$ ，其中  $w \in \mathbb{R}^{K \times N}$ ， $b \in \mathbb{R}^K$ 。
2. 对于每个样本  $(x^{(i)}, y^{(i)})$  和每个类别  $k$ ，计算偏导数（梯度）：

$$g_k^{(i)}(x^{(i)}) = \sum_{j=1}^n w_{kj} x_j^{(i)} + 1 \quad k = 1, 2, \dots, K$$

$$g_k^{(i)}(x^{(i)}) = g_k^{(i)}(x^{(i)}) - w_{y^{(i)}k} x^{(i)} \quad k = y^{(i)}$$

3. 检查样本  $x^{(i)}$  是否违反了 KKT 条件。如果违反，解决子问题，通过在单纯形上投影来更新权重  $w_k$  和偏置  $b_k$ 。
4. 重复步骤 2 和 3，直到达到预定的迭代次数或损失函数的改进小于某个阈值（即收敛）。
5. 对于给定的实例  $x$ ，计算每个类别的决策函数值：

$$\hat{y} = \arg \max_k \left( \sum_{j=1}^n w_{kj} x_j + b_k \right)$$

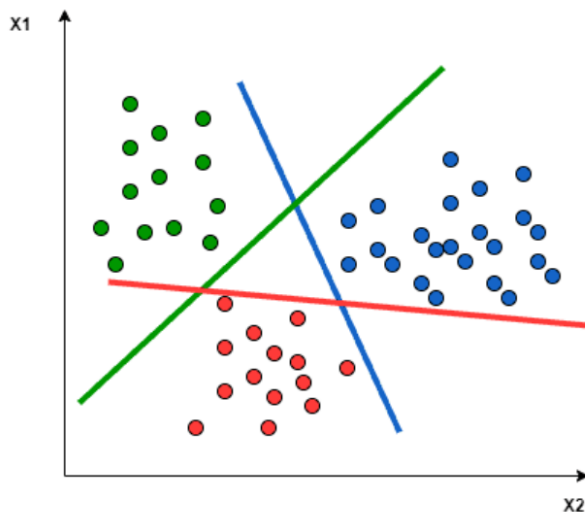


图 2.3: MSVM 分类示例

## 3 实验环境与平台

### 3.1 硬件环境

- 处理器为: AMD Ryzen 7 7840H w

### 3.2 测试环境

- VScode:1.85
- Python3:3.9.18
- jupyter\_core:5.7.1

### 3.3 依赖的 python 相关包

- sklearn: 机器学习包, 本项目中需要其中的功能性模块, 如切分数据集所需的模块
- pandas: 处理数据集
- numpy: 使用其中的 array 用于计算
- matplotlib: 用于可视化数据



## 4 结果与分析

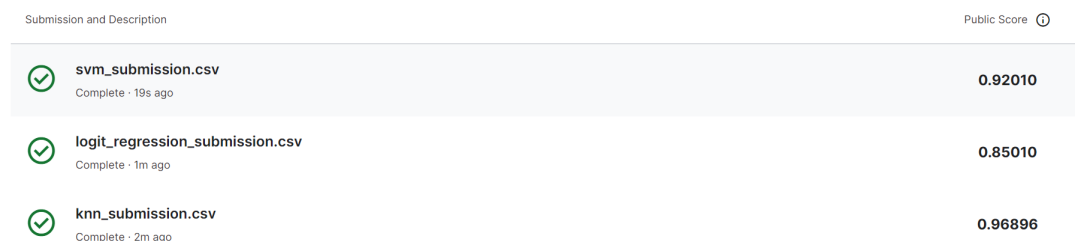
### 4.1 算法测试与结果评估

将 KNN、MLR 与 MSVM 模型预测的结果与已知准确率极高的结果比较（该结果准确率在 99.7% 以上），得到的结果如下表所示：

表 4.1: 算法测试结果

模型	正确率
KNN	97.05 %
MLR	85.13 %
MSVM	92.22 %

再将预测结果保存在三个 csv 文件中，并在 kaggle 网站上提交测试，得到的 score 如下图所示：






Submission and Description	Public Score
 <b>svm_submission.csv</b> Complete · 19s ago	<b>0.92010</b>
 <b>logit_regression_submission.csv</b> Complete · 1m ago	<b>0.85010</b>
 <b>knn_submission.csv</b> Complete · 2m ago	<b>0.96896</b>

图 4.1: kaggle 上三种模型测试结果

### 4.2 结果分析

三种方法中，MLR 算法的效果最差，KNN 的效果最好，MSVM 的效果也不错。本人在查证了逻辑回归分析部分文献资料的基础上，认为模型表现不佳的原因可能有以下几点：

1. 多元逻辑回归假设特征与类标签之间存在线性关系，而手写数字图像通常包含复杂的模式和非线性特征，线性边界可能不足以将不同类别有效分开。缺乏非线性变换的能力使得逻辑回归在处理复杂数据时表现不佳。
2. 手写数字图像的原始像素值不足以有效地表示数据的结构和模式。多元逻辑回归直接使用原始像素值作为特征，这种表示方式难以捕捉图像的局部和全局特征。
3. 训练数据集中不同数字的样本数量不平衡，逻辑回归无法充分学习所有类别的特征，导致分类性能下降。

KNN 模型因为其简单性，我们不作过多的分析，只是 KNN 模型的运行时间过长，将近 100 分钟，这与实验所用设备核心处理器性能有关。如果能将训练数据与训练模型有效迁移至 GPU 上，模型的运行时间或将大大缩短。

下面我们来探讨一下 MSVM 算法的表现相对优良的原因。

1. 支持向量机通过使用核函数将输入数据映射到更高维的特征空间。在高维空间中，即使是线性不可分的数据也有可能变得线性可分。因此，模型可以完整把握手写数字图片的特征。
2. 支持向量机的目标是找到一个能够最大化类别间边界（margin）的超平面。因此，SVM 可以更好地泛化到未见过的数据，即便训练数据中存在一些噪声和异常值。
3. 正则化参数  $C$  控制了分类边界的平滑度和平衡误差与边界的权重。这有效地防止了模型的过拟合，从而提升其泛化能力。

基于以上分析，我认为想要进一步优化识别手写数字图像问题，可以从以下几个方面着手：

1. 对于 KNN 模型，我们可以尝试使用加权 KNN 模型，给距离较近的邻居赋予更大的权重，提高预测的准确性。
2. 对于 MSVM 模型，我们可以换用 One-vs-One 的多分类策略，对于  $N$  个类别，使用两两组合的方式，构建  $\frac{N(N-1)}{2}$  个二元分类器。
3. 使用卷积神经网络（CNN）等更为复杂深度学习模型，可以更高效做出预测。
4. 集成现有多个模型，将多个模型的预测结果进行组合，如集成 KNN 和 SVM 的预测结果，提高最终的分类效果。

## 5 个人体会

经过本次项目，我对于图像识别有了更加深刻的理解。通过查阅各种资料，甚至是看 CS231n(著名计算机视觉课程)，我了解了各种用以处理图像任务的模型，并亲笔跟着讲义进行数学推导。在整个学习的过程中，结合课程中学到的知识，以及实际工程上实现的过程，我对于各种模型的建立，以及各种可调节参数的调节方法都有了更深入的了解。

通过这个项目，我感受到机器学习这一领域浩如烟海，如今各个学科领域都多少与机器学习交叉，想要真正在机器学习领域有所建树，不仅要多实战，更要努力提升自己的数学理论水平。在查阅资料时，我就常常苦恼于各种复杂的数学表达（尤其是在学习支持向量机的凸优化问题时）。同时，我们应该熟练使用 python 进行各种数据处理，所谓熟能生巧，多用才能牢记。相信在理论与实战的双重磨练下，我们在机器学习领域的能力一定能得到大幅提高，为从事更复杂数据工作与数据研究夯实基础。

## 参考文献

- [1] 周志华. 机器学习: 第 3、6、10 章. 清华大学出版社, 2016.
- [2] Dan Jurafsky、James H. Martin. Speech and Language Processing (3rd ed. draft): chapter 5.
- [3] Fei-Fei Li. Deep Learning for Computer Vision Course notes.