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：数据科学与大数据技术

班　　级：BD2202

学　　号：U202215641

姓　　名：曲睿

成　　绩：

指导教师：张腾

完成日期：2024 年 5 月 20 日

目 录

1 课题任务

1.1 选题简介

选择**参考选题二**：肥胖风险的多类别预测。具体为基于个体性别、年龄、身高、体重等特征判断肥胖程度的**多特征多分类学习问题**。

1.2 任务要求

a) 分类准确率在 **0.8** 以上；

b) 模型训练需自己手动实现，**严禁**直接调用已经封装好的各类机器学习库，但可以使用 NumPy 等数学运算库或功能性方法。

2 算法设计与实现

2.1 数据预处理

在进行机器学习任务前，对数据集进行适当的分析与处理是必不可少的过程，恰到好处的预处理不仅有助于选择高效的模型，而且能显著提高预测准确性。

2.1.1 数据预览

本数据集包括训练用数据 train.csv 与测试用数据 test.csv，使用 Pandas 库中 read_csv 方法将 csv 文件读取为 DataFrame 结构，并使用 head 方法显示前 5 行数据，其中部分列如图 2.1 所示。

	id	Gender	Age	Height	Weight	family_history_with_overweight	FAVC	FCVC	NCP	CAEC
0	0	Male	24.443011	1.699998	81.669950	yes	yes	2.000000	2.983297	Sometimes
1	1	Female	18.000000	1.560000	57.000000	yes	yes	2.000000	3.000000	Frequently
2	2	Female	18.000000	1.711460	50.165754	yes	yes	1.880534	1.411685	Sometimes
3	3	Female	20.952737	1.710730	131.274851	yes	yes	3.000000	3.000000	Sometimes
4	4	Male	31.641081	1.914186	93.798055	yes	yes	2.679664	1.971472	Sometimes

图 2-1 训练用数据预览效果

2.1.2 特征与标签分析

对列名与数据格式进行解读可得如下表所示的特征与标签分析。

表 2-1 特征一览

特征名	数据类型	含义
Gender	object②	性别
Age	float	年龄
Height	float	身高(m)

特征名	数据类型	含义
Weight	float	体重(kg)
fhwo①	object	家庭成员是否肥胖
FAVC	object	是否经常高热量饮食
FCVC	float	进食蔬菜的频率
NCP	float	一天进餐次数
CAEC	object	两餐之间进食情况
SMOKE	object	是否吸烟
CH2O	float	饮水量(L)
SCC	object	是否检测每日卡路里
FAF	float	一天运动次数
TUE	float	电视观看时间
CALC	object	饮酒频率
MTRANS	object	出行方式
NObeyesdad	object	肥胖程度

①fhwo：“family_history_with_overweight”的简写。

②object：即字符串类型

根据上表可将特征按离散或连续量进行初步分类，如 Gender、fhwo、FAVC、CAEC、SMOKE、SCC、CALC、MTRANS 属于**离散特征**，Age、Height、Weight、FCVC、NCP、CH2O、FAF、TUE 属于**连续特征**，而 NObeyesdad 为**标签**。对于这两类特征与标签，将采用不同的处理方法。

2.1.3 连续特征预处理

对于连续特征，主要的处理方式是**归一化**，以去除量纲不同的影响，也可以考虑将某几个特征**用一个特征进行综合**。具体操作如下：

a) 考虑到 BMI 是一项综合了身高与体重的指标，能够一定程度地反映身体肥胖程度，于是**使用 BMI 代替 Height、Weight**，计算公式如下。

$$BMI = \frac{Weight}{Height^2} \tag{2-1}$$

b) 归一化有多种方式，主要包括 min-max 标准化与 Z-score 标准化，需根据数据分布形式进行选择。对 Age 特征绘制分布图，如图 2-2 所示，观察到有**少量离群值**，其他连续特征的分布中同样存在，而 min-max 标准化受离群值影响较大，因此选择 **Z-score 标准化**处理连续特征。

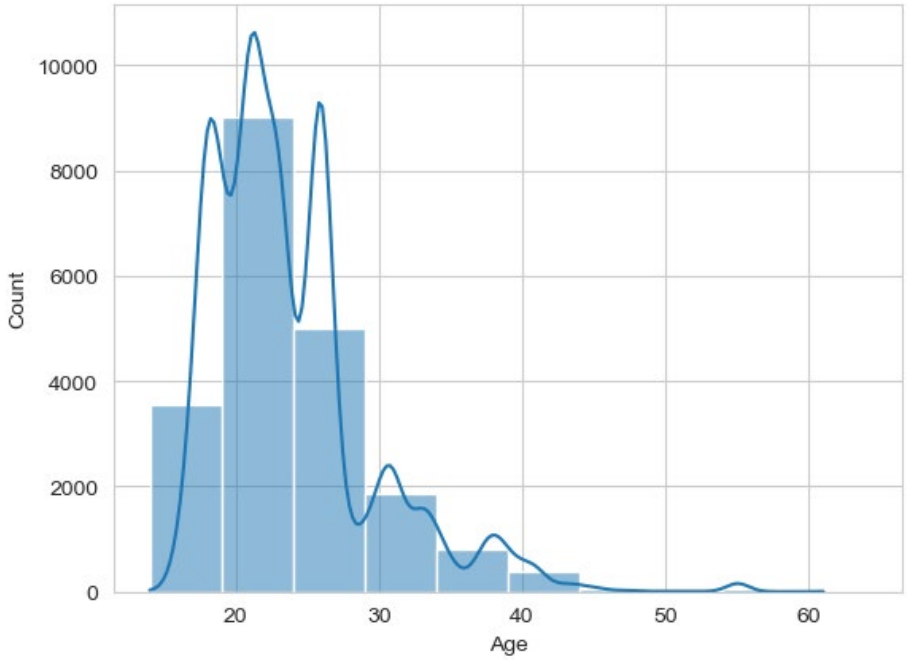

图 2-2 Age 特征分布图

2.1.4 离散特征与标签预处理

对于离散特征与类别标签，主要的处理方式是**编码**，如果类别本身**没有强相关性**，还需**独热编码**。但此数据集中的离散特征各有特点，需采取不同的策略。

a) Gender、fhwo、FAVC、SMOKE、SCC 均属于**二类**离散特征，且两个取值间**没有强相关性**，因此直接进行 **0/1 编码**即可；

b) CAEC、CALC 属于**四类**离散特征，但四个取值分别为 no、Sometimes、Frequently、Always，显然在刻画频率，具有**偏序**关系。但需注意，**训练用数据中 CALC 的取值并无 Always**。考虑上述两点，有两种方法处理 CALC，一是采取朴素的[0, 1]间**均匀映射**，四个取值分别映射为 0、1/3、2/3、1，此时无需考虑 CALC 取值的缺失；二是进行独热编码，并添加“CALC=Always”特征列，但这种做法并未利用取值之间的强相关关系，平添多个无谓的特征。衡量利弊后，此处选择**方法一**进行处理。

c) MTRANS 属于**五类**离散特征，且取值间**没有强相关性**，因此对其进行**独热编码**。

d) 标签 NObeyesdad 共有七类，根据肥胖程度由轻到重映射为 0 至 6 之间的整数。

2.1.4 预处理结果

预处理后的数据预览如图 2-3 所示，包含 **19 个特征**。

	Age	BMI	CAEC	CALC	CH2O	FAF	FAVC	FCVC	Gender	MTRANS_0	MTRANS_1
11790	-0.498150	-0.190991	0.333333	0.333333	-0.048348	0.598798	1	-0.836259	0	0	0
10495	-0.499608	-0.690196	0.333333	0.333333	-1.691823	0.021774	1	-0.836259	1	0	0
2554	-0.675414	-0.873868	0.666667	0.333333	-0.048348	2.407549	1	1.039146	0	0	0
6478	-0.851221	-1.447477	0.666667	0.333333	-1.691823	-1.171113	0	1.039146	0	0	0
9256	-1.027027	-1.103176	0.333333	0.000000	-0.048348	-1.171113	1	-0.836259	1	0	0

MTRANS_2	MTRANS_3	MTRANS_4	NCP	SCC	SMOKE	TUE	family_history_with_overweight
0	1	0	-2.497017	0	0	-1.024319	1
0	1	0	0.338356	0	0	-1.024319	0
0	1	0	0.338356	0	0	-1.024319	1
0	1	0	0.338356	0	0	-1.024319	0
0	1	0	0.338356	0	0	0.636498	0

图 2-3 预处理结果预览

2.1.5 划分训练集与测试集

使用 sklearn 库中的 train_test_split 方法将训练用数据划分为训练集与测试集，其中**测试集占 20%**。

2.2 模型一：基于随机梯度更新的 SoftMax 回归

Logistic 回归是一种简单的二分类算法，**SoftMax 回归**是其一般形式，用于多分类。假设输入数据$\{(X_1, y_1), (X_2, y_2), \dots, (X_m, y_m)\}$有 k 个类别，即$y \in \{1, 2, \dots, k\}$，**SoftMax 回归的任务是计算输入数据 X_i 归类于每一类的概率**，即公式 2-2 所示。

$$h_w(X_i) = \begin{bmatrix} p(y_i = 1|X_i; w) \\ p(y_i = 2|X_i; w) \\ \vdots \\ p(y_i = k|X_i; w) \end{bmatrix} = \frac{1}{\sum_{j=1}^{k} e^{w_j^T X_i}} \begin{bmatrix} e^{w_1^T X_i} \\ e^{w_2^T X_i} \\ \vdots \\ e^{w_k^T X_i} \end{bmatrix} \quad (2-2)$$

其中 w 是模型的参数矩阵，w_{ij} 表示特征 i 在计算类别为 j 的概率时的权重，概率 p 通过 SoftMax 变换得到，如公式 2-3 所示。

$$p(y_i = j|X_i; w) = SoftMax\left(w_j^T X_i\right) = \frac{e^{w_j^T X_i}}{\sum_{l=1}^{k} e^{w_l^T X_i}} \quad (2-3)$$

对上述公式进行分析，设$z = WX$，此时的 z 实质上是高维空间内的一个超平面，且由于没有偏置项 bias，其必过空间原点。因此我们需**加入偏置项 bias**，使得 z 这一决策超平面可以出现在空间内的任何位置，对 z 进行 SoftMax 映射变

换即可得到概率向量 y，如公式 2-4 所示。

$$y = SoftMax(z) = SoftMax(WX + bias) \qquad (2-4)$$

算法原理图如图 2-4 所示。

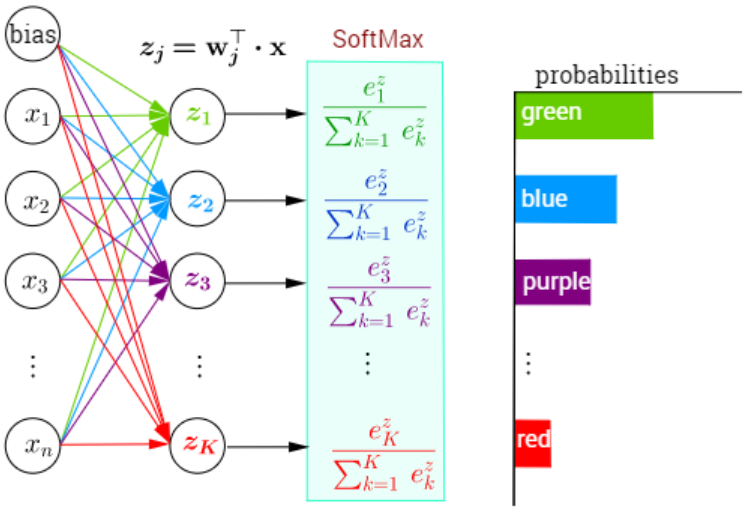

图 2-4 SoftMax 回归原理图

模型参数通过**梯度下降法**逐次更新，对于诸如 Logistic 回归、SoftMax 回归的线性分类器，**梯度均为$\hat{y} - y$**，实现时可省去求导的过程。为加快更新速度，使用**随机梯度更新**，每轮进行**随机取样**，将取样规模设为超参数以进行调整，计算梯度并更新参数。算法流程图如图 2-5 所示。

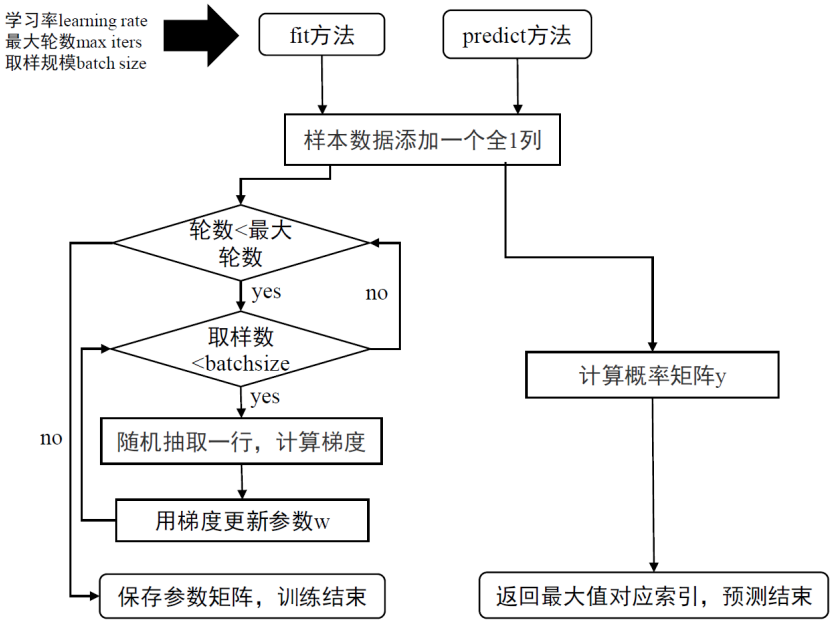

图 2-5 SoftMax 回归算法流程图

该模型的超参数有**学习率**(Learning rate, LR)、**最大轮数**(Max iters)、**取样规**

模(Batch size)。显然，每轮的训练时间完全取决于取样规模的大小，而总训练时间取决于最大轮数与取样规模的乘积，训练时间的长短相当于训练使用样本量的大小。

学习率在调参中相当重要，而在该模型中，**学习率与取样规模互相限制**，因为当**取样规模较小**时，噪音会相对突出，此时应使用**较小的学习率**“小心翼翼”地调整参数。经测试，超参数分别为 **0.01、500、1000** 时预测效果最好。

2.3 模型二：基于 KD-Tree 优化的 K-NN 算法

K-NN 是一种简单的多分类算法，朴素 K-NN 算法无需训练，可直接进行预测，原理不再赘述。但显然，朴素 K-NN 的求解基于**线性扫描**，需进行测试集-训练集-特征的三重循环求解，**数据量很大时求解效率很低**。对于本数据集而言，进行一次预测需花费约 $15000*15000*20/10^7$=**4500 秒**，复杂度不可接受。

因此选择使用 **KD-Tree** 进行优化。KD-Tree 中存储的是 k 维数据，在训练的过程中不断用垂直于坐标轴的超平面将 k 维空间划分，构成一系列 k 维超矩形区域，**训练的具体过程**为：

a) 构造根结点，令 j=1，根据第 j 个维度的特征进行排序，取**中位数**作为结点，将中位数左右两边的数据递归构造左右子树；
b) 重复：对于深度为 depth 的结点，选择第 j=(j mod k)+1 个特征进行排序与划分，同样去中位数作为结点，将中位数左右两边的数据递归构造左右子树；
c) 没有数据可以划分时终止算法。

对样本进行**预测的具体过程**为：

a) 从根结点出发，不断与当前结点特征值进行比较，小于则进入左子树，大于则进入右子树（相当于不断进行**二分查找**）；
b) 直至进入叶结点，将其标记为当前最邻近的点，距离为 s_{min}；
c) 回溯，若当前结点总距离小于 s_{min}，则更新最邻近的点与距离，若当前结点特征距离小于 s_{min}，则进入当前结点的另一子结点进行比较；
d) 回退至根结点时终止算法。

需注意 KDTree 中的 **k 是特征数**，与 K-NN 中的超参数 k 完全不同，因此基于 KDTree 优化的 K-NN 算法实际上并**无参可调**，即**其预测结果完全固定**。

2.4 模型三：基于信息增益的 ID3 决策树

决策树也是一种简单的多分类算法，其构建过程对应着**特征空间的划分**。首先构建根结点，将全部数据放在根结点，选择一个**最优特征**，根据这一特征对根结点数据进行划分，进入不同的子结点，由此递归构建子结点，直至达到叶结点，点内数据根据标签进行投票并将叶结点归类于票数最多的标签。

最优特征的选择有多种方式，此处选择**信息增益最大**的特征。多类别向量 X 中第 i 个类别 x_i 的**信息量 $I(x_i)$**定义如公式 2-5 所示，其中 $p(x_i)$表示数据集中类别为 x_i 的数据出现的概率。

$$I(x_i) = -\log_2 p(x_i) \tag{2-5}$$

信息熵 H(X)即信息量的期望，定义如公式 2-6 所示。

$$H(X) = \sum_{x \in X} p(x) I(x) \tag{2-6}$$

条件熵 H(Y|X)表示特征 X 为某个值时类别为 Y 的熵，计算如公式 2-7 所示。

$$H(Y|X) = \sum_{x \in X} p(x) \sum_{y \in Y} p(y|x) I(y|x) \tag{2-7}$$

特征 A 对训练集 D 的**信息增益 g(D, A)**计算如公式 2-8 所示。

$$g(D, A) = H(D) - H(D, A) \tag{2-8}$$

预测时只需递归访问决策树直至叶结点，返回其所属标签即可。

决策树的超参数主要有**树的最大深度**。深度过小的决策树**欠拟合**，不能充分利用数据集特征；深度过大则**过拟合**，即使在测试集上表现良好，在测试用数据上依然会出现准确率较低的预测。经测试，**最大深度为 4** 时拟合效果最好。

3 实验环境与平台

3.1 实验环境

实验环境如表 3-1 所示。

表 3-1 实验环境

名称	配置信息
操作系统	Windows 11
开发语言	Python 3.11.5
CPU	Intel Core i7-12700H

内存	16GB

3.2 实验平台

实验平台为 Jupyter Notebook。

4 结果与分析

4.1 模型评估

采用的评估指标如表 4-1 所示。

表 4-1 评估指标一览

指标	目的
准确率	反映模型正确性
F1 score	反映模型稳健性
混淆矩阵	直观展示模型性能
训练/预测时间	反映模型运行速度

4.1.1 模型一 SoftMax 回归评估

调用 sklearn 库中相关评估方法计算**准确率与 F1 score**，而在 F1 score 的计算中，需选择使用**宏平均或微平均**，而选择依据为预测结果中各类别数量差异与关注度。因此首先对训练用标签进行绘图分析，如图 4-1 所示，可以观察到肥胖 III 型最多，超重 I 型最少，且**数量差异较大**。而对于本课题而言，目标是判断肥胖类型，显然应当更**注重数量最多的肥胖 III 型**，因此采用**宏平均**计算 F1 score。

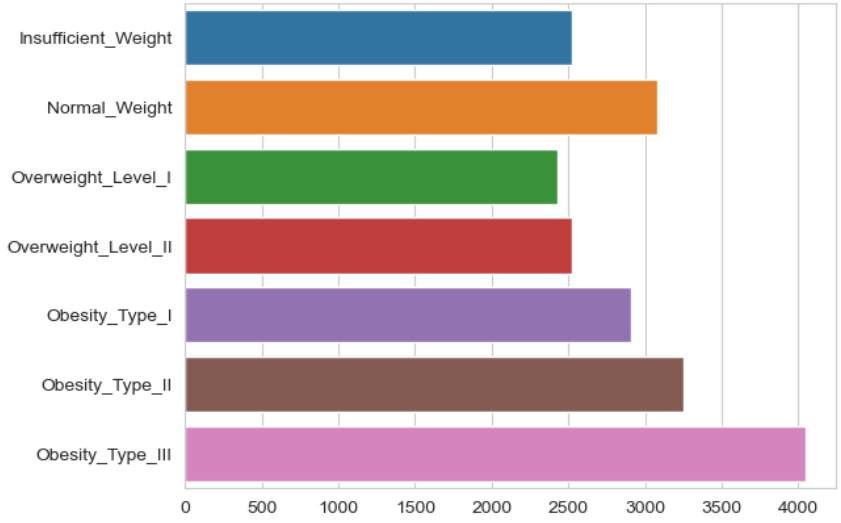

图 4-1 类别数量对比图

为横向对比自己实现的 MyLogRegClf 模型，调用 sklearn 库中的 LogisticRegression 进行训练与预测，并同样计算指标进行对比，结果如表 4-2 所

示。

表 4-2 自实现模型一与 sklearn 性能对比

指标	MyLogRegClf	sklearn.LogisticRegression
准确率	0.8372	0.8471
F1 score	0.8177	0.8300
训练时间/s	18.9	1.7

可以观察到，自己实现的模型和 sklearn 中模型在控制**超参数相同**的情况下相比，**准确率与 F1 score 略低**，但该模型的**正确性与稳健性基本符合要求**，可以进一步预测测试用数据；而训练时间相比 sklearn 中模型**慢了 10 倍以上**，显然自己实现的模型在训练的**时间复杂度上还有很大提升空间**。

使用测试集绘制**混淆矩阵**如图所示，可以观察到模型在**肥胖 III 型**上的**表现最好**，基本全部判断正确；其次是肥胖 II 型；在超重 II 型上的表现最差，近一半的超重 II 型样本被归类至超重 I 型与肥胖 I 型。一个可能的原因是，就人的直觉而言，肥胖体型显然易于区分，而正常体重与略微超重之间的界限并不明显，不能“一眼”认出，也即**单纯的线性分类器不足以完全区分**。

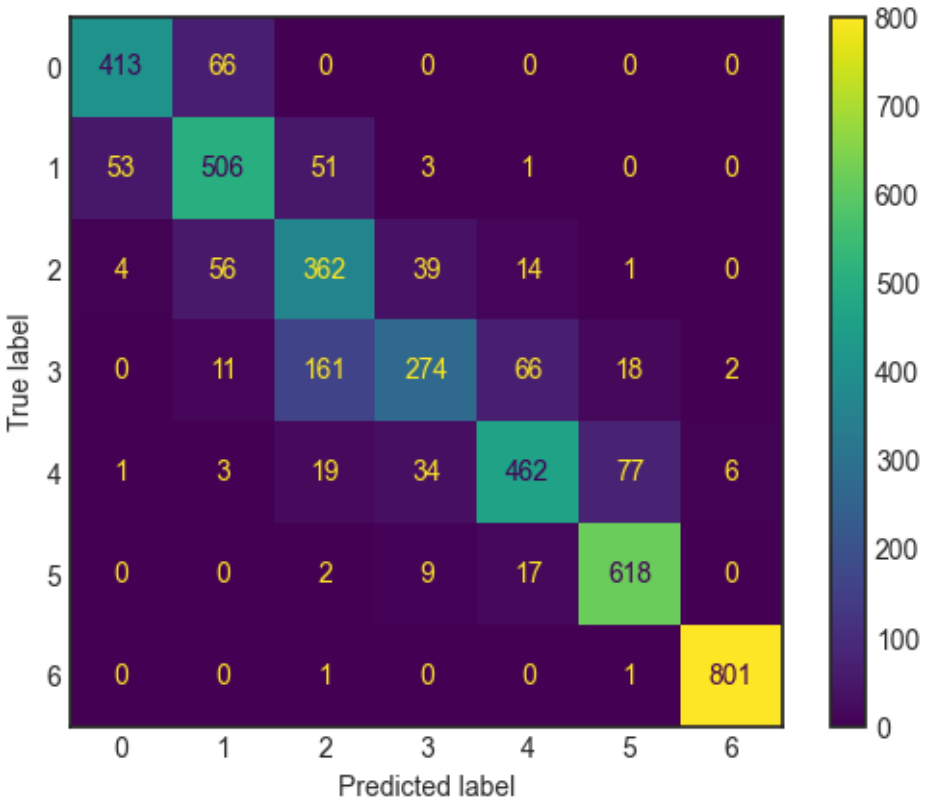

图 4-2 测试集混淆矩阵图例

4.1.2 模型二 KD-Tree 评估

同样调用 sklearn 库中的 KNN 模型，并指定参数 algorithm 为 kd_tree，进行指标对比，如表 4-3 所示。

表 4-3 自实现模型二与 sklearn 性能对比

指标	MyKdTreeClf	sklearn.Knn(with KDTree)
准确率	0.7486	0.7599
F1 score	0.7232	0.7336
预测时间/s	46.6	0.7

可以观察到，自己实现的模型和 sklearn 中模型在控制使用算法（KD-Tree）相同的情况下相比，**准确率与 F1 score 略低**，但均未突破 0.8，没有满足准确率的要求；预测时间上差别仍然很大，可见在实现效率上 sklearn 依然更优。

两个模型的准确率与 F1 score 差别不大说明自己实现的 KdTree 模型在原理上是正确的，据此可推测预测未满足要求的主要原因是**模型本身具有缺陷**。

K-NN 算法，包括 KD-Tree 算法主要依靠预测点周围有限的邻近的样本，而本数据集具有**维数高、类别不平衡**的特点，在进行距离计算时，一是采用的**欧式距离并不识别方向**，二是**高维度易引入误差**，两者叠加使得 K-NN 与 KD-Tree 算法的**预测准确率不尽人意**。

4.1.3 模型三决策树评估

同样调用 sklearn 库中的决策树模型，进行指标对比，如表 4-4 所示。

表 4-4 自实现模型三与 sklearn 性能对比

指标	MyDecTreeClf	sklearn.DecTreeClf
准确率	0.8410	0.8408
F1 score	0.8268	0.8266
训练时间/s	51.4	0.0

可以观察到，自己实现的模型和 sklearn 中模型在控制超参数相同的情况下相比，**准确率与 F1 score 略高**，正确性与稳健性基本符合要求，可以进一步预测测试用数据；但预测时间上差别仍然很大。

4.2 预测结果

经本地模型评估可发现，**模型一 SoftMax 回归**与**模型三决策树**的准确率初步满足要求。使用这两个模型分别对测试用数据 test.csv 进行预测，并将结果上传至 Kaggle 线上测评，测评结果如图 4-所示，分数分别为 **0.84040** 与 **0.83255**，

均已满足准确率要求。

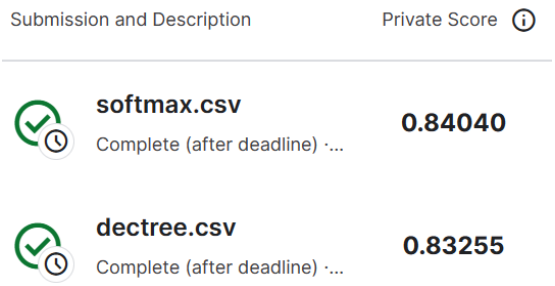

图 4-3 线上测评结果

4.3 模型分析

将三个模型的准确率、F1 score、训练或预测所用时间综合绘制为柱状图，如图 4-4 所示。

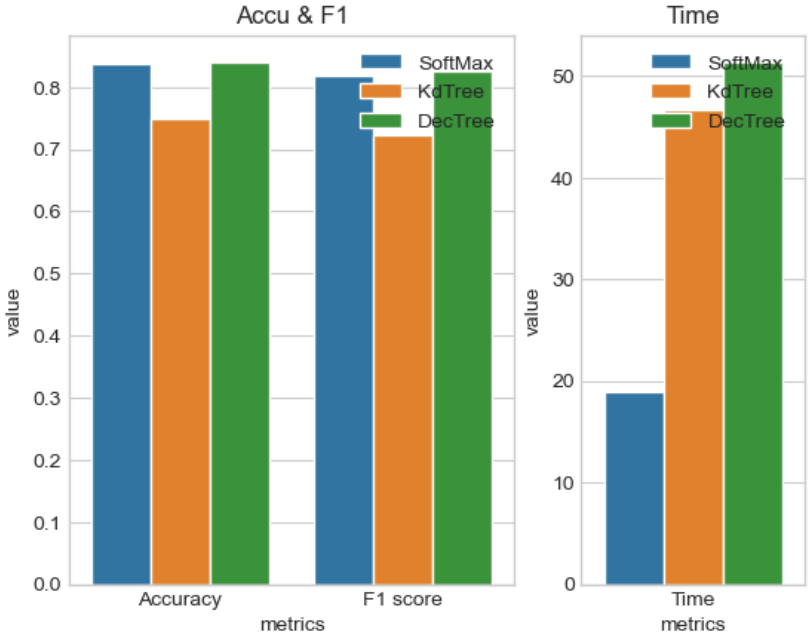

图 4-4 三个自实现模型性能对比图

可以观察到，**SoftMax 回归**在模型的正确性、稳健性上表现良好，所用时间明显优于其他两个模型，**综合表现最佳**；**决策树**同样展现了良好的正确性与稳健性，但**所用时间最长，效率较低**；而 **KdTree** 在三方面的表现**均不佳**。据此将三个模型各自优劣与特点总结为表 4-5。

表 4-5 模型优劣一览

模型	正确性	稳健性	训练效率	缺点
SoftMax	优	优	高	形式简单，难以拟合真实非线性数据
KD-Tree	良	良	低	难以处理高维数据
决策树	优	优	低	易过拟合，训练耗时过长

5 个人体会

完成这项课题的过程总体而言是困难的，但我从中收获了不少新知。

数据预处理对我而言不算陌生，但从开始时一股脑全部独热编码到报告中呈现的预处理结果之间，我也走过不少弯路，画相关系数热力图、删除弱相关特征、min-max 归一化、Z-score 标准化、尝试将离散值转换为连续值而非单纯地编码……虽然在探索的过程中，我了解了很多未曾见过的处理方法，但归根结底，预处理的最好依据还是数据集本身的特点，单纯地套用高级的预处理方法很可能是负优化。

自己实现算法一定是课题中最煎熬的部分。不仅是因为原理的复杂、矩阵计算的繁琐，更是因为对 Python 与 NumPy 等数学运算库几近陌生，看着关于矩阵维度、形状等等的报错信息却无从下手甚至网上搜索不到时，心中只有绝望。所幸，机器学习算法在国内的论坛中是很热门的，从逻辑回归到神经网络，从决策树到随机森林，从 K-NN 到支持向量机，我从各路学者的文章中学到了许多，从各种生动形象的讲解中了解了许多模型的原理，尝试过自己实现逻辑回归、决策树、KD-Tree、神经网络、支持向量机、随机森林，并最终完成了报告中所提到的三个模型的手动实现。在实现模型前的选择阶段，我调用 sklearn 中各类模型进行预测，发现我想实现的模型准确率险过要求线时不禁汗颜，“那我实现的模型怎么可能过 0.8”，但当我看到手动实现的准确率与 sklearn 不分上下时，更加有成就感了，虽然在运行效率上还有很大提升空间。

在评估模型与撰写报告时，我更多地尝试用图解代替文字，以达到更加直观的讲解或展示作用，为此我学习了许多 matplotlib 与 seaborn 画图方法，这也是我第一次在报告中尽量压缩字数与内容。

这是我第一次使用自己构建的机器学习方法解决数据分析与预测问题。在选修这门课程前，我对机器学习还抱有“神奇”、“魔幻”的刻板印象，一行 sklearn 代码就能秒出模型，预测准确率高达 99.99%.....而当我敲打着结课报告的最后一部分时，我依然觉得机器学习是很困难、很高深的领域，但我也觉得它不再那么神秘，在模型训练的五十秒内，整个微积分、线代、概率论都在为我闪烁。