实验一 KNN分类

58119216 刘幽远

问题概述

- 利用KNN算法对输血服务中心数据集中的测试集进行分类。

表1 输血服务中心数据集信息

样例数量	特征维度	特征类型	类别数量
798	4	数值	2

- **任务说明**
  - **任务一：** 利用欧式距离、切比雪夫距离、曼哈顿距离作为KNN算法的度量函数对测试集进行分类。实验报告中，要求分析三种距离度量在该数据集上的优劣同时，要求在验证集上分析近邻数k对KNN算法分类精度的影响。
  - **任务二：**利用马氏距离作为KNN算法的度量函数，对测试集进行分类。马氏距离是一种可学习的度量函数，定义如下:

    $d_M(x_i, x_j) = \sqrt{(x_i - x_j)^T \mathbf{M}(x_i - x_j)}$

    其中，$\mathbf{M} \in \mathbb{R}^{d \times d}$是一个半正定矩阵，是可以学习的参数，由于$\mathbf{M}$的半正定性质，可以将上述定义表述为：

    $d_M(x_i, x_j) = \sqrt{(x_i - x_j)^T \mathbf{A^T A}(x_i - x_j)} = ||Ax_i - Ax_j||_2$

    其中，矩阵$\mathbf{A} \in \mathbb{R}^{e \times d}$，马氏距离可以理解为对原始特征进行线性映射，然后计算欧氏距离。

    给定以下目标函数，在训练集上利用梯度下降法对马氏距离进行学习：

    $\max_{\mathbf{A}} f(\mathbf{A}) = \max_{\mathbf{A}} \sum_{i=1}^{N} \sum_{j \in C_i} p_{ij}$

    其中，C_i表示与样例x_i同类的样例集合，p_{ij}定义为：

    $$p_{ij} = \begin{cases} \frac{exp(-d_M(x_i,x_j)^2)}{\sum_{k \neq i} exp(-d_M(x_i,x_k)^2)}, j \neq i \\ 0, j = i \end{cases}$$

任务一

实验过程

- 首先注意到实验数据中有一维数据量级远超其他三维，所以做一次归一化，但是并不是约束在$(0,1)$内，而是改为约束在(0,100)内，防止可能的精度问题
- 定义三种Metric
- 然后进行建模，KNN算法使用K-D Tree进行计算
  - K-D Tree是一种特殊的二叉树，用于解决一系列高维空间中的问题
  - K-D Tree的构建依赖于数据点的方差，对于数据点的划分，选择当前方差最大的一维，然后将中位数点作为当前根节点，划分左右子树
  - 使用K-D Tree计算KNN的过程如下：

        - 构造优先队列Q（最大优先队列，距离从小到大的顺序排队）
        - 在树中找出包含目标点x的叶结点：从根结点递归访问K-D Tree，若x当前维的坐标小于根节点的坐标，则移动到左子结点，否则移动到右子结点，直到子结点为叶结点
        - 将此节点插入队列Q，如果Q长度大于k，则删除队尾元素
        - 递归地向上回退，在每个结点进行以下操作
            - 如果该结点保存的实例点比L队首更近，则将此节点插入队列Q，如果Q长度大于k，则删除队尾元素
            - 检查该子结点的父结点的另一子结点对应的区域是否有比L队首更近的点。

        > 具体地，检查另一子结点对应的区域是否与以“Q队尾结点”为球心、以目标点与“Q队尾结点”间的距离为半径的超球体相交，如果相交，可能在另一个子结点对应的区域内存在距离“Q队尾结点”更近的点，移动到另一个子结点．递归地进行k近邻搜索，如果不相交，向上回退

        - 当回退到根结点时，搜索结束，最后的Q队列中的结点即为x的k近邻点
- 定义评判标准
    - $Acc = Accuracy = \frac{number\ of\ correct}{total}$
- 对比结果
    - 设定不同的K值
    - 绘制三种Metric在K变化时的曲线
    - 根据曲线的比较结果决定最优解

实验结果与分析

- 结果分析
    - 从K值来看，随着K的递增，三种$Metric$下的Acc都逐渐下降

- 同时，在大多数情况下，曼哈顿距离与欧式距离优于切比雪夫距离
- 在引入F_1_score（单独计算进行比较）之后，欧氏距离更胜一筹
- 经过分析数据，私以为，本题的数据只有4个维度，所以适用于低维数据的欧式距离更胜一筹
- 伴随着数据维度的提高，计算复杂度和表达都更简洁的曼哈顿距离才能显现出其威力

任务二

实验过程

- 因为使用了梯度下降法，所以需要先进行梯度计算公式的推导

假设$d_{ij} = -exp(-d_M(x_i, x_j)^2$ ，那么有：

$p_{ij} = \frac{d_{ij}}{\sum_{k \neq i} d_{ik}}$

$\frac{\partial d_{ij}}{\partial A} = -2d_{ij}A(x_i - x_j)(x_i - x_j)^T$

那么：

$\frac{\partial p_{ij}}{\partial A} = \frac{1}{(\sum_{k \neq i} d_{ik})^2}(\frac{\partial d_{ij}}{\partial A}\sum_{k \neq i} d_{ik} - d_{ij}\sum_{k \neq i}\frac{\partial d_{ik}}{\partial A})$

代入化简后可得

$\frac{\partial p_{ij}}{\partial A} = -2p_{ij}A((x_i - x_j)(x_i - x_j)^T - \sum_{k \neq i}(x_i - x_k)(x_i - x_k)^T)$

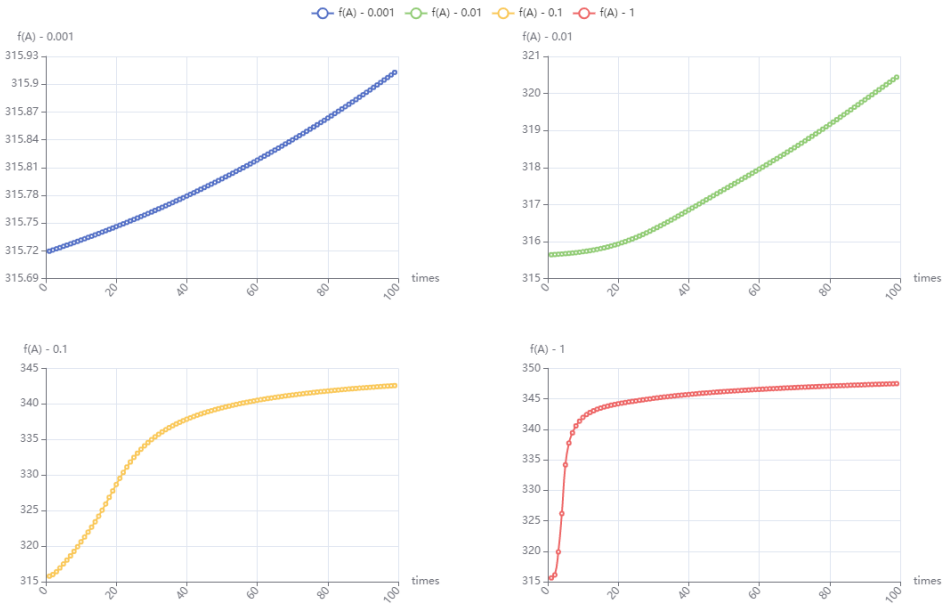
目标梯度为：

$\frac{\partial f}{\partial A} = \sum_{i=1}^{N}\sum_{j \in C_i}\frac{\partial p_{ij}}{\partial A} = -2A\sum_{i=1}^{N}\sum_{j \in C_i} p_{ij}((x_i - x_j)(x_i - x_j)^T - \sum_{k \neq i} p_{ik}(x_i - x_k)(x_i - x_k)^T)$

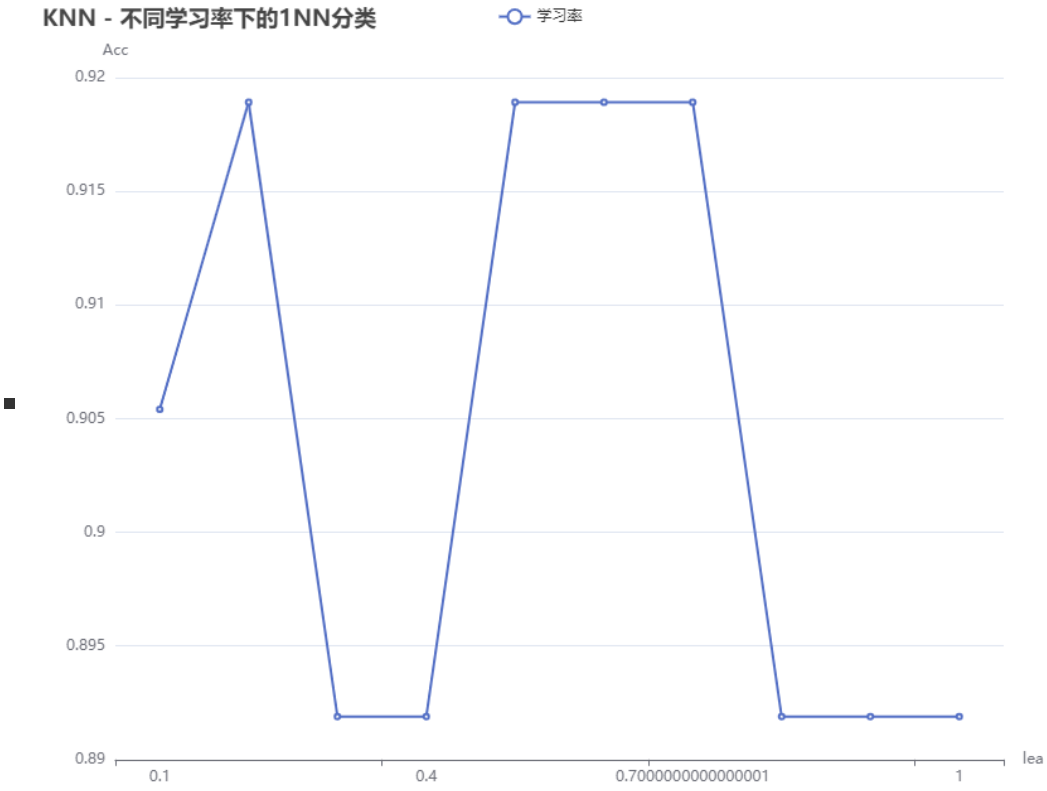
- 推导完成之后，根据本题的具体要求，进行梯度下降算法的改进，包括但不限于
  - 因为求最大值，所以改-为+
  - 学习率范围划定
- 进行计算

实验结果与分析

- 学习率的选择，首先需要直接观察目标函数的变化趋势来进行粗略的学习率划分
- 第一轮筛选选择的学习率为$1e-3, 1e-2, 1e-1$和1来进行测试



    - 显然，学习率为$1e-1$和1是我们需要的曲线，那么接下来的筛选就在这之间进行
- 第二轮筛选选择$0.1, 0.2, \ldots 1$的学习率，将矩阵A按照公式计算距离，代入不同的K比较趋势
    - 因为均为$K=1$时结果最优，所以不做图片展示
    - 进一步的测试就是比较$K=1$时谁更优，对比图如下：



    - 能够看出在$0.5, 0.6, 0.7$之间，准确率达到了峰值
    - 将学习率设置为0.6时，将按K增长的变化图线加入任务一的对比图如下：

    - 显然，学习得到的马氏距离优于任何一种固定的距离计算方式

- 分析：
    - 首先，f函数的收敛值大约在345左右，选择$0.1-1$的学习率都能在100次训练左右收敛
    - 训练得到的A矩阵表现非常优秀
    - 无论是固定距离计算方式还是马氏距离，都呈现出随着K的增长识别率下降的趋势，可以推测数据具有充分的复杂性，K更小时准确率更高

总结

- 本次的实验难度较大，本次实验也是我第一次实际使用K-D Tree解决KNN问题，事实证明KNN确实有高效解决该问题的能力，并没有因为我使用的语言是单线程且效率较低的Javascript语言而受到影响
- 马氏距离的学习部分我还是使用的python，因为涉及到了复杂的矩阵计算，js性能不足的缺点被极大的放大
- 对于实验结果，虽然感觉非常不可思议，并且发现valid数据集与train数据集存在重合数据，但是并不清楚这一现象属于需要自己排查的问题还是数据本身就是如此，因为题目并没有说明重复的数据是否可视为对特征的强化，如果不是，那这属于我的疏漏，对于相关的知识还是不够了解
- 使用js而非python的原因是我本人对于python的数据可视化库，如matplotlib等还是不够熟悉，所以选择使用更熟悉的js来进行可视化，虽然效果还不错，但是之后仍然需要加强相关方面的学习
- 卡的时间最久的是梯度下降的推理，数理基础仍然是我的薄弱点
- 总体来说，是一次收获颇丰的实验