实现 LSH+kNN 模型分类星体并对其超参数研究

利用主动学习神经网络学习机器学习中超参数的最佳设定

李星汉[1] 王东来[2] 吴曦[3] 余宸林[4]

[1234] 物理科学与工程学院
同济大学

2024 年 1 月 3 日

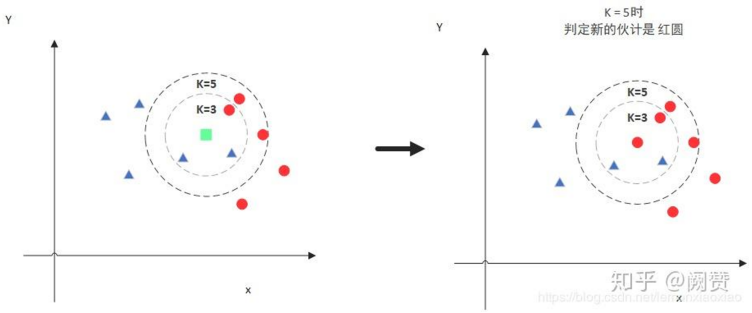
1. LSH+kNN 算法实现并实现天体物理分类
   - kNN 与 LSH+kNN
   - 天体物理：小行星分类

2. 对哈希桶数（超参数）设定的研究
   - 最小稳定除数 (最大稳定桶数)：w
   - 主动学习确定超参数

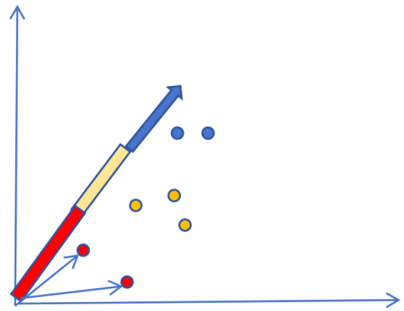
1. LSH+kNN 算法实现并实现天体物理分类
   - kNN 与 LSH+kNN
   - 天体物理：小行星分类
2. 对哈希桶数（超参数）设定的研究
   - 最小稳定除数 (最大稳定桶数)：w
   - 主动学习确定超参数

kNN 与 LSH+kNN

- kNN——k 个最近邻邻居: 根据最近邻 k 个节点确定该节点类型
- LSH——局部敏感哈希: 给每组数据加上一个 label



图



图

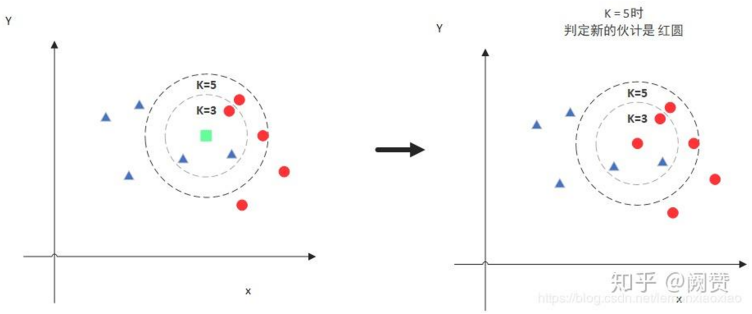
最近邻判断: 根据欧式距离

根据欧氏距离，近的分入一个桶

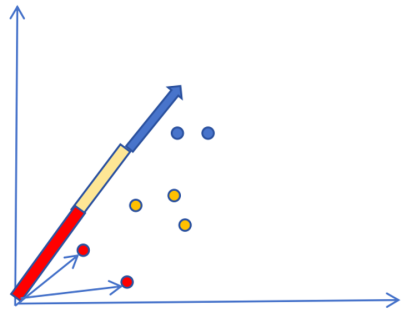
- 原本的 kNN 需要遍历所有数据点，LSH+kNN 只在分好哈希值后，只需要遍历哈希值相同的所有点

kNN 与 LSH+kNN

- kNN——k 个最近邻邻居: 根据最近邻 k 个节点确定该节点类型
- LSH——局部敏感哈希: 给每组数据加上一个 label



图



图

最近邻判断: 根据欧式距离

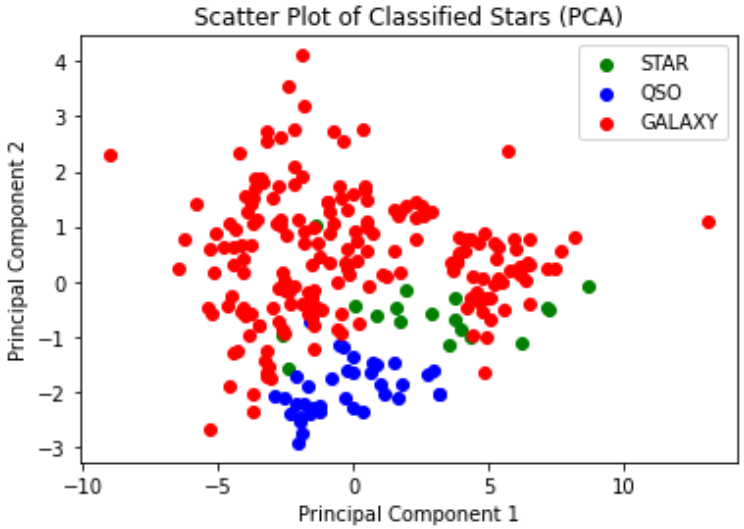
根据欧氏距离，近的分入一个桶

- 原本的 kNN 需要遍历所有数据点，LSH+kNN 只在分好哈希值后，只需要遍历哈希值相同的所有点 ——在保证一定精度的情况下提高效率

根据天体光谱分类小行星：LAMOST 数据集

- 数据特征：3690-9100 埃的波长范围内的一系列辐射强度值
- 天体类别：恒星、星系、类星体

- 分类后的示意图



图

- 对于大数据集，kNN 运行时间显著降低 (对比.py)!

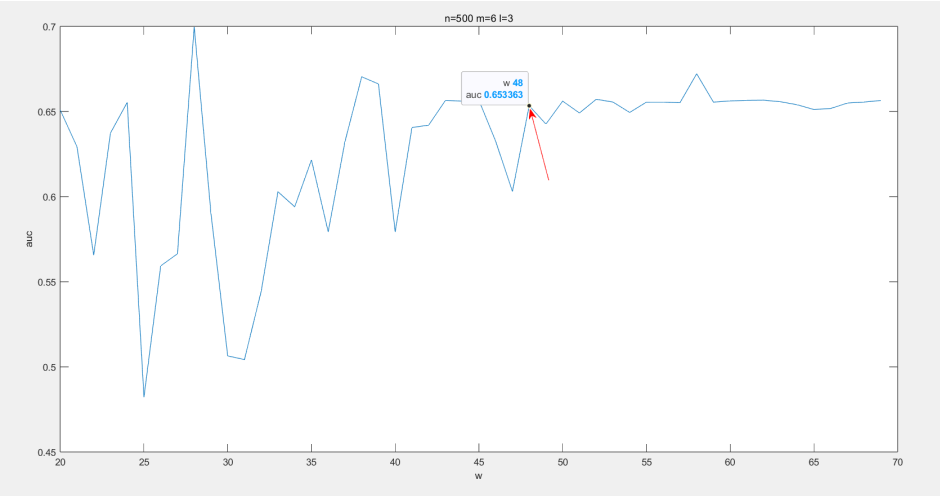
```
In [112]: runfile('C:/Users/Administrator/D
Desktop/人工智能/期末项目')
time1: LSH-kNN 0.26906299591064453
time2: kNN without LSH 7.600545644760132
```

1. LSH+kNN 算法实现并实现天体物理分类
   - kNN 与 LSH+kNN
   - 天体物理：小行星分类

2. **对哈希桶数（超参数）设定的研究**
   - **最小稳定除数 (最大稳定桶数)：w**
   - **主动学习确定超参数**

最小稳定除数 (最大稳定桶数)：w

随着哈希桶数的减小，程序运行效率会减慢，程序精度会提高。
给定不同 w 值，观察 auc 的变化并绘制图像



图：n=500,m=6,l=3 情况下，AUC 值随 w 的变化

——随着 w 增大（哈希桶数减小），auc 值趋于稳定

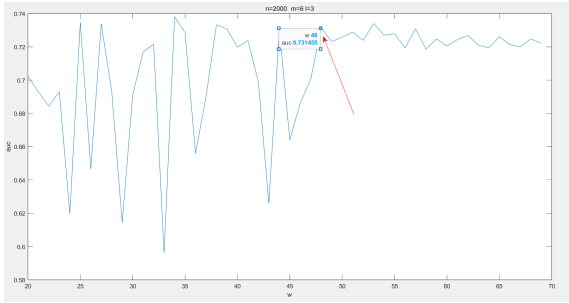
- 最小稳定除数 (Minimum maximum w)——LSH+kNN 算法精度与效率的平衡点：我们发现，程序精度提高有一个临界值。一开始 w 很小时，AUC 值非常不稳定，随着 w 值的增大，AUC 趋于稳定。我们提出：把程序趋于稳定的临界点称为**最小稳定除数**（以下简称稳定 w）

1. LSH+kNN 算法实现并实现天体物理分类
   - kNN 与 LSH+kNN
   - 天体物理：小行星分类
2. 对哈希桶数（超参数）设定的研究
   - 最小稳定除数 (最大稳定桶数)：w
   - 主动学习确定超参数

- 超参数 w 的设定：能不能通过对不同数据集稳定 w 的学习，实现通过用户给定数据类别确定超参数——在保证精度的前提下最大程度提高效率
- 与 w 有关的数据集特征：
  - 数据个数 n
  - 标签个数 m
  - 数据类别 l

- 超参数 w 的设定：能不能通过对不同数据集稳定 w 的学习，实现通过用户给定数据类别确定超参数——在保证精度的前提下最大程度提高效率
- 与 w 有关的数据集特征：
  - 数据个数 n
  - 标签个数 m
  - 数据类别 l
  - 标签与数据时相关性：w=1 时的 AUC，在 0.4∼0.8 内时，我们认为符合我们研究的数据特征

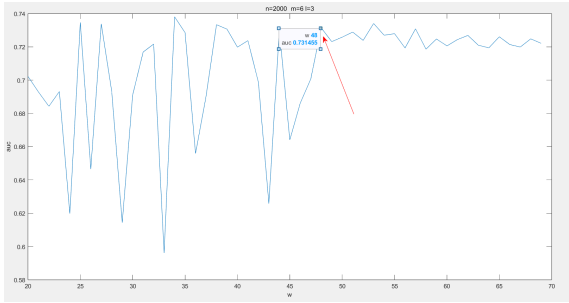
- 然而，对于每一个 w 的获取，需要跑海量的 w 来确定达到稳定的临界点



图

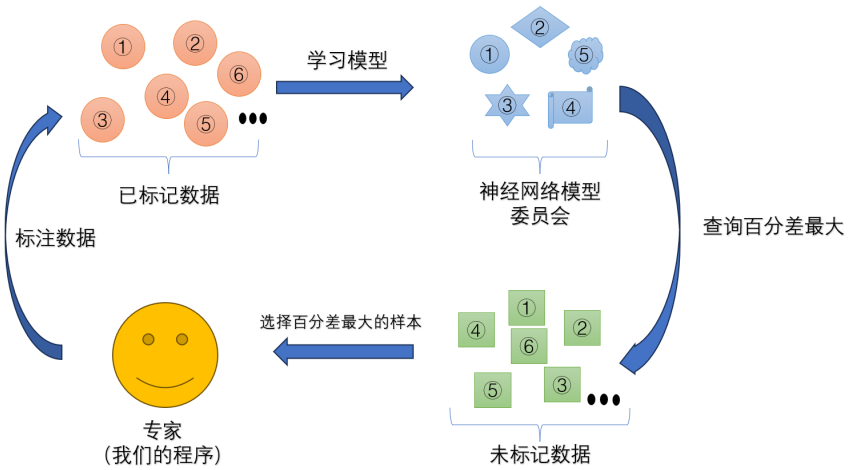
- 超参数 w 的设定：能不能通过对不同数据集稳定 w 的学习，实现通过用户给定数据类别确定超参数——在保证精度的前提下最大程度提高效率
- 与 w 有关的数据集特征：
  - 数据个数 n
  - 标签个数 m
  - 数据类别 l
  - 标签与数据时相关性：w=1 时的 AUC，在 0.4∼0.8 内时，我们认为符合我们研究的数据特征

- 然而，对于每一个 w 的获取，需要跑海量的 w 来确定达到稳定的临界点



图

——主动学习：在数据量明显不够时，通过对分类效果最差数据的“刻意学习”，提高



图：主动学习思想

- 训练集：我们之前已经得到的 n，m，l，稳定 w
- 模型委员会：我们设定委员（模型）个数为 5
- 判定模型效果是否真的有提升：数据平均百分差

主动学习实操

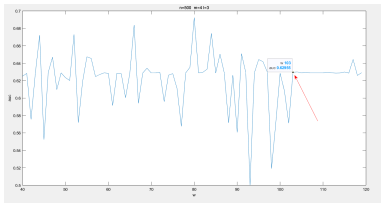
- 第一次测试返回平均百分差，百分差最大的一组

```
In [77]: runcell(0, 'C:/Users/Administrator/D
第一次训练平均百分差为： 301.1676720210484
百分差最大的一组nml组合: [500   4   3]
```

图

第一次测试平均百分差为：301

- 将百分差最大的测试集“专家标定”（放到我们的程序里面再跑一遍）



图：得到稳定 w 为 103

- 将稳定 w=103 与先前的 nml 值重新作为一组参数进行学习

```
#重新训练模型
new_nml = nml_samples[max_diff_idx]
new_w = np.array([[103]])  # 用实际测量的w值替换
```

图

- 继续测试，得到百分差为

```
In [90]: runcell('主动学习第二次', 'C:/Users
第二次训练平均百分差为： 62.32717037200928
百分差最大的nml组合的: [550   6   3]
```

图

第二次测试平均百分差为：62

——发现百分差确实有降低

不足与展望

- kNN+LSH 速度的提升：我们速度的提升只是对比了我们用相同逻辑，但哈希桶只有一个的‘kNN’算法，然而，和 scipy 库中的 kNN，甚至自己编写的 kNN，速度都没有明显提升。这和我们调用了太多循环可能有关，通过对 python 数据结构更深刻的认识，我们可以把 LSH+kNN 代码写得更好
- 主动学习参数的选择：实际上，一个数据集的特征不但与 n，m，l 有关，更与这个数据集的标签与这个数据集的类别的相关程度有关（比如，一个水果是不是西瓜，和大小的关系，远小于和颜色的关系），这直接关系到我们模型底层逻辑是否。我们只粗略地用 w=1 时的 auc 排除了相关性太强或太弱的数据集，实际上，可以把 w=1 时的 auc 或其他能反映数据标签与类别相关度的参数当作主动学习的参数。
- 稳定 w 的选择：我们只是粗略的选择看起来稳定的临界点，但是没有量化的标准。下一步，我们可以采用 YOLOV8 算法训练找出稳定 w 的算法，实现全流程自动化封装。
- 主动学习数据：主动学习数据量仍然太少，如果我们可以封装成包，

小组分工

- LSH+kNN 代码：李星汉，吴曦
- 主动学习代码：王东来，吴曦
- 数据集寻找与处理：余宸林