机器学习算法 kNN 实现简介

k 近邻（k-Nearest Neighbor，简称 kNN）学习是一种常用的监督学习方法。所谓"近朱者赤，近墨者黑"。在分类任务中（本项目是这样），为了判定未知样本的类别，以全部训练样本作为代表点，基于某种距离度量，计算未知样本与所有训练样本的距离，并以近邻 k 个样本的大多数类别作为决策未知样本类别的依据，即选择这 k 个样本中出现最多的类别标记作为预测结果，这种方法也被形象地称为“投票法”。

在回归任务中时使用"平均法"，即将这 k 个样本的实值输出标记的平均值作为预测结果（有待在新的版本中实现）。

与其他的机器学习方法相比，训练只是将样本数据保存起来，等到收到测试样本再进行处理。由此，它看起来没有明显的训练过程，是“懒惰学习”的著名代表。

顾名思义，k 无疑是一个十分重要的参数，通常来说，k 取奇数。当 k 取不同值时，分类结果会有显著不同。如右图，考虑待测样本（用圆形表示）的类别，当 k=3 时，测试样本被判定为三角形；当 k=5 时，测试样本被判定为矩形。而且，k 对噪声数据敏感。

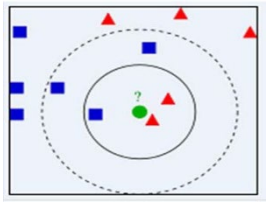

图 1 – k 的选取

由此，项目中评价算法采用了多次使用不同的参数 k 评价数据集的预测正确性，每一次计算从整个数据集中随机抽样，多次得到的结果取 precision 最大的 k 值作为结果。本项目的相应算法可由以下伪代码表示（具体实现详见代码 eval_runner.cpp ）：

```
SETTRAININGANDTESTSET(dataset, range)
1   trainset ← ‹ ›
2   testset ← ‹ ›
3   start ← RANDOM(1, 5)
4   index ← 0
5   while  index < LENGTH(dataset)
6   do  if  start%(LENGTH(dataset)/range) = 0
7           then  INSERT(testset, dataset[index])
8           else  INSERT(trainset, dataset[index])
9       start ← start+1
10      index ← index+1
11  return  trainset, testset
```

图 2 – 抽象的随机抽样算法

```
EVALUATE(dataset, range)
1   result_list ← ‹ ›
2   k ← 1
3   while   k < 20
4   do   trainset, testset ← SETTRAININGANDTESTSET(dataset, range)
5        precision ← GETPRECISION(k, trainset, testset)
6        INSERT(result_list, ‹k, precision›)
7        if   FIND(result_list, k) = 1
8             then   return ‹k, precision›
9        k ← k+2
10  INVERTEDSORTBYPRECISION(result_list)
11  return   result_list[0]
```

图 3 - 抽象的评价算法

另一方面，若采用不同的距离计算方式，则找出的"近邻"可能有显著差别，从而也会导致分类结果有显著不同。

为解决这一问题，项目实现了两种距离计算方法（有待在更新的版本中增加新的距离算法）：欧氏距离和余弦相似度。在文件 kNNHelper.h 中定义宏 _DISTANCE_TYPE 进行区分，0 表示欧氏距离（默认），1 表示余弦相似度，可以根据需要在编译前修改。

张昊
Holger Zhang
2020-05-03