第3章

k-近邻算法

本章目标

- 理解 k-近邻算法的数学思想；
- 掌握实现 k-近邻算法所需要的一般手段，包括 k 值的选取、距离的度量和快速检索；
- 实现简单的 k-近邻算法，并自主对比不同参数下的表现。

k-近邻（k-Nearest Neighbor，KNN）算法是一种常用的分类或回归算法。给定一个训练样本集合 D 以及一个需要进行预测的样本 x，k-近邻算法的思想非常简单：对于分类问题，k-近邻算法从所有训练样本集合中找到与 x 最近的 k 个样本，然后通过投票法选择这 k 个样本中出现次数最多的类别作为 x 的预测结果；对于回归问题，k-近邻算法同样找到与 x 最近的 k 个样本，然后对这 k 个样本的标签求平均值，得到 x 的预测结果。k-近邻算法的描述如算法 3-1 所示。

算法 3-1　k-近邻算法

输入：训练集 $D=\{(\boldsymbol{x}_1,y_1),(\boldsymbol{x}_2,y_2),\cdots,(\boldsymbol{x}_m,y_m)\}$；$k$ 值；待预测样本 x；如果是 k-近邻分类，同时给出类别集合 $C=\{c_1,c_2,\cdots,c_K\}$

输出：样本 x 所属的类别或预测值 y

1. 计算 x 与所有训练集合中所有样本之间的距离，并从小到大排序，返回排序后样本的索引

$$P=\underset{i}{\operatorname{argsort}}\{d(\boldsymbol{x},\boldsymbol{x}_i)\mid i=1,2,\cdots,m\}$$

2. 对于分类问题，投票挑选出前 k 个样本中包含数量最多的类别

$$\text{Return } y=\underset{i=1,2,\cdots,K}{\operatorname{argmax}}\sum_{p\in P}I(\boldsymbol{x}_p=c_i)$$

3. 对于回归问题，用前 k 个样本标签的均值作为 x 的估计值

$$\text{Return } y=\frac{1}{k}\sum_{p\in P}y_p$$

对 k-近邻算法的研究包含三方面：k 值的选取、距离的度量和如何快速地进行 k 个近邻的检索。

3.1 k 值的选取

投票法的准则是少数服从多数,所以当 k 值很小时,得到的结果就容易产生偏差。最近邻算法是这种情况下的极端,也就是 $k=1$ 时的 k-近邻算法。最近邻算法中,样本 x 的预测结果只由训练集中与其距离最近的那个样本决定。

如果 k 值选取较大,则可能会将大量其他类别的样本包含进来,极端情况下,将整个训练集的所有样本都包含进来,这样同样可能会造成预测错误。一般情况下,可通过交叉验证、在验证集上多次尝试不同的 k 值来挑选最佳的 k 值。

3.2 距离的度量

对于连续变量,一般使用欧氏距离直接进行距离度量。对于离散变量,可以先将离散变量连续化,然后再使用欧氏距离进行度量。

词嵌入(Word Embedding)是自然语言处理领域常用的一种对单词进行编码的方式。词嵌入首先将离散变量进行独热(one-hot)编码,假定共有 5 个单词$\{A,B,C,D,E\}$,则对 A 的独热编码为$(1,0,0,0,0)^{\mathrm{T}}$,B 的独热编码为$(0,1,0,0,0)^{\mathrm{T}}$,其他单词类似。编码后的单词用矩阵表示为

$$\boldsymbol{X}=\begin{array}{c}\begin{matrix}A & B & C & D & E\end{matrix}\\\begin{pmatrix}1&0&0&0&0\\0&1&0&0&0\\0&0&1&0&0\\0&0&0&1&0\\0&0&0&0&1\end{pmatrix}\end{array} \tag{3-1}$$

随机初始化一个用于词嵌入转化的矩阵 $\boldsymbol{M}_{d\times 5}$,其中每一个 d 维的向量表示一个单词。词嵌入后的单词用矩阵表示为

$$\boldsymbol{E}=\boldsymbol{M}_{d\times 5}\boldsymbol{X}=\begin{array}{c}\begin{matrix}A & B & C & D & E\end{matrix}\\\begin{pmatrix}x_{11}&x_{12}&x_{13}&x_{14}&x_{15}\\x_{21}&x_{22}&x_{23}&x_{24}&x_{25}\\\vdots&\vdots&\vdots&\vdots&\vdots\\x_{d1}&x_{d2}&x_{d3}&x_{d4}&x_{d5}\end{pmatrix}\end{array} \tag{3-2}$$

矩阵 $\boldsymbol{E}$ 中的每一列是相应单词的词嵌入表示,d 是一个超参数,$\boldsymbol{M}$ 可以通过深度神经网络(见第 11 章)在其他任务上进行学习,之后就能用单词词嵌入后的向量表示计算内积,用以表示单词之间的相似度。对于一般的离散变量同样可以采用类似词嵌入的方法进行距离度量。

3.3 快速检索

当训练集合的规模很大时,如何快速找到样本 x 的 k 个近邻成为计算机实现 k-近邻算法的关键。一个朴素的思想是:

(1) 计算样本 x 与训练集中所有样本的距离。

(2) 将这些点依据距离从小到大进行排序选择前 k 个。

算法的时间复杂度是计算到训练集中所有样本距离的时间加上排序的时间。该算法的第(2)步可以用数据结构中的查找序列中前 k 个最小数的算法优化,而不必对所有距离都进行排序。一个更为可取的方法是为训练样本事先建立索引,以减少计算的规模。kd 树是一种典型的存储 k 维空间数据的数据结构(此处的 k 指 x 的维度大小,与 k-近邻算法中的 k 没有任何关系)。建立好 kd 树后,给定新样本后就可以在树上进行检索,这样就能够大大降低检索 k 个近邻的时间,特别是当训练集的样本数远大于样本的维度时。关于 kd 树的详细介绍见书末参考文献[3]。

3.4 实例:基于 k-近邻算法实现鸢尾花分类

本节以鸢尾花(Iris)数据集的分类来直观理解 k-近邻算法。为了在二维平面展示鸢尾花数据集,这里使用花萼宽度和花瓣宽度两个特征进行可视化,如图 3-1 所示。

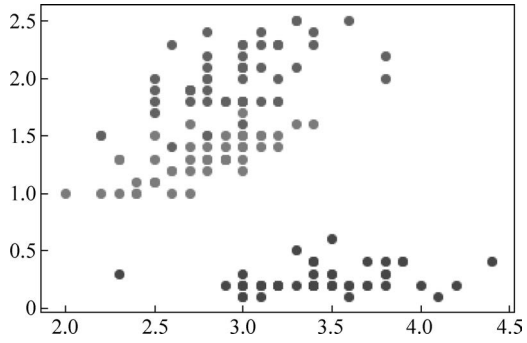

图 3-1 鸢尾花数据集(见彩插)

图中蓝色数据点表示 Setosa,橙色数据点表示 Versicolour,绿色数据点表示 Virginica。sklearn 中提供的 k-近邻模型称为 KNeighborsClassifier,代码清单 3-1 给出了模型构造和训练代码。

代码清单 3-1 k-近邻模型的构造与训练

```
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier as KNN
if __name__ == '__main__':
    iris = load_iris()
    x_train, x_test, y_train, y_test = train_test_split(
        iris.data[:, [1,3]], iris.target)
    model = KNN()
    model.fit(x_train, y_train)
```

代码清单 3-2 对上述模型进行了测试。根据程序输出可以看出,模型在训练集上的准确率达到 0.964,在测试集上的准确率达到 0.947。

代码清单 3-2　模型测试

```
train_score = model.score(x_train, y_train)
test_score = model.score(x_test, y_test)
print("train score:", train_score)
print("test score:", test_score)
```

图 3-2 展示了模型的决策边界。可以看出,几乎所有样本点都落在相应的区域内,只有少数边界点可以落在边界外。

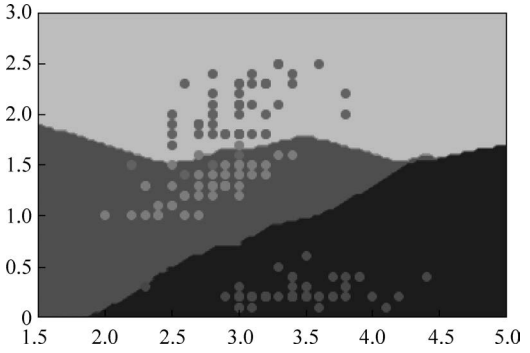

图 3-2　k-近邻模型的决策边界(见彩插)

k-近邻模型默认使用 $k=5$。当 k 过小时,容易产生过拟合现象;当 k 过大时,容易产生欠拟合现象。图 3-3 展示了 k 为 1 或 50 时的决策边界。不难看出,当 $k=1$ 时,决策边界更加复杂;而当 $k=50$ 时,决策边界较为平滑。

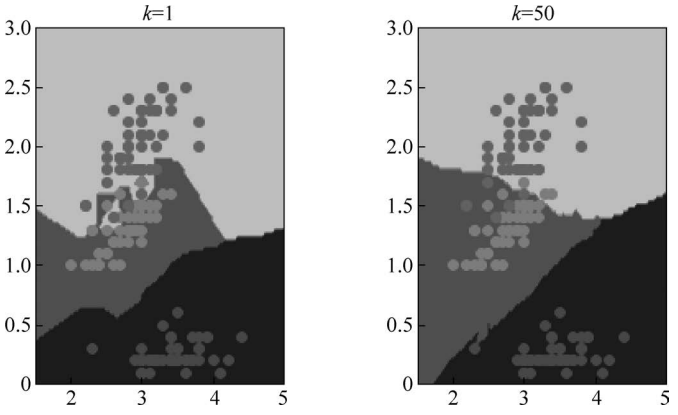

图 3-3　不同 k 值对决策边界的影响(见彩插)


题库

习题 3

一、选择题

1. 下面关于 KNN 算法说法正确是(　　)。

A. KNN 算法的时间复杂度是 $O(nkt)$,其中 k 为类别数,t 为迭代次数

B. KNN 算法是一种非监督学习算法

C. 使用 KNN 算法进行训练时，训练数据集中含有标签

D. K 值确定后，使用 KNN 算法进行样本训练时，每次所形成的结果可能不同

2. 如图 3-4 所示，以下(　　)是 KNN 算法的训练边界。

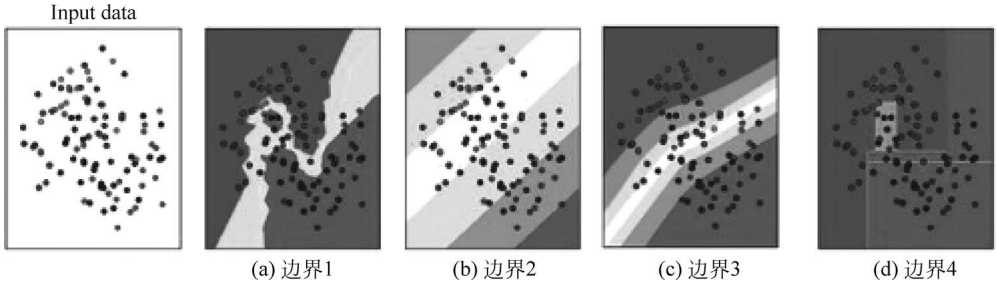

图 3-4　决策边界可视化

A. 图 3-4(b)　　B. 图 3-4(a)　　C. 图 3-4(d)　　D. 图 3-4(c)

3. 以下关于 KNN 算法的描述，不正确的是(　　)。

A. KNN 算法只适用于数值型的数据分类　　B. KNN 算法对异常值不敏感

C. KNN 算法无数据输入假定　　D. 以上说法都不正确

4. 使用 $K=1$ 的 KNN 算法，图 3-5 二类分类问题+和∘分别代表两个类，那么，用仅拿出一个测试样本的交叉验证方法，交叉验证的错误率是(　　)。

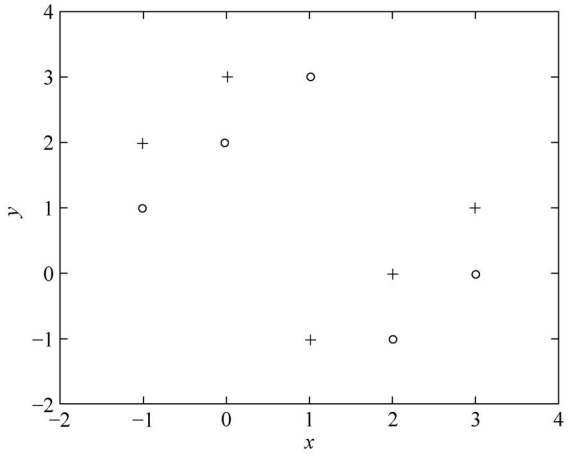

图 3-5　二类分类问题

A. 0　　B. 100%　　C. 0 到 100%　　D. 以上都不是

5. 一般情况下，KNN 算法在(　　)情况下效果最好。

A. 样本呈现团状分布　　B. 样本呈现链状分布

C. 样本较多但典型性不好　　D. 样本较少但典型性好

二、判断题

1. KNN 算法较适用于样本容量较大的类域的自动分类。　　(　　)

2. KNN 算法以空间中 K 个点为中心进行聚类。 ( )

3. 随着 K 值的增大,决策边界会越来越光滑。 ( )

4. KNN 算法适合解决高维稀疏数据上的问题。 ( )

5. 相对 3 近邻模型而言,1 近邻模型的 Bias 更大,Variance 更小。 ( )

三、填空题

1. 在 KNN 模型中,超参数 K 的选择对模型的表现有较大的影响。一般而言,对比 1 近邻模型和 3 近邻模型,1 近邻模型的 Bias 更_____,Variance 更_____。

2. KNN 算法是_____学习算法。

3. KNN 算法对_____不敏感。

4. KNN 算法可以用来解决_____问题。

5. KNN 中若 $K=1$,样本 x 的预测结果只与由训练集中与其_____距离最近的那个样本决定。

四、问答题

1. 简述 KNN 算法中 K 值对结果的影响。

2. 简述 KNN 算法的优点。

3. 简述 KNN 算法的缺点。

4. 简述 KNN 算法中如何选取 K 值。

5. 简述 KNN 算法的原理。

五、应用题

1. 根据表 3-1 所示数据集数据通过身高、体重、鞋子尺码,预测数据为[176,71,38]的样例的性别。

表 3-1 服饰数据集示例

身 高	体 重	鞋子尺码	性 别
170	65	41	男
166	55	38	女
177	80	39	女
179	80	43	男
170	60	40	女
170	60	38	女

2. 通过手写数字的数据集进行数字识别,数据集通过 https://github.com/angleboygo/data_ansys/blob/master/knn.rar 获取。