《人工智能导论》

实验八：KNN 算法

学　号 22920212204396

姓　名 黄子安

2024 年 5 月 29 日

实验八： KNN 算法

229202212204396 黄子安

一、实验目的

K 最近邻 (k-Nearest Neighbors，KNN) 算法是一种分类算法，1968 年由 Cover 和 Hart 提出，可以应用于字符识别、文本分类、图像识别等领域。

该算法的思想是：一个样本与数据集中的 k 个样本最相似，如果这 k 个样本中的大多数属于某一个类别，则该样本也属于这个类别，是最简单易懂的机器学习算法之一。本实验通过解决 iris 数据集分类，来更好的熟悉和掌握 KNN 算法。

二、实验内容

使用 iris 数据集进行 KNN 实验。

iris 数据集的中文名是安德森鸢尾花卉数据集，英文全称是 Anderson's Iris data set。iris 包含 150 个样本，对应数据集的每行数据。每行数据包含每个样本的四个特征和样本的类别信息，所以 iris 数据集是一个 150 行 5 列的二维表。

通俗地说，iris 数据集是用来给花做分类的数据集，每个样本包含了花萼长度、花萼宽度、花瓣长度、花瓣宽度四个特征（前 4 列），我们需要建立一个分类器，分类器可以通过样本的四个特征来判断样本属于山鸢尾、变色鸢尾还是维吉尼亚鸢尾，iris 的每个样本都包含了品种信息，即目标属性（第 5 列，也叫 target 或 label），从而实现监督学习。

三、实验过程

先定义一个方法用于计算样本之间的欧几里得距离，这里也可以采用曼哈顿距离、切比雪夫距离等等

```
def euclidean_distance(x1, x2):
    return np.sqrt(np.sum((x1 - x2) ** 2))
```

之后根据算法的原理编写算法类，计算测试样本与其余所有样本的距离，之后选择最近的 k 个邻居，选择出现最多的邻居类别作为自己的预测值，之后使用 most_common 选出出现次数最多的邻居，返回一个元组，其中每一个元素是一个键值对，表示对应的标签和出现次数，通过 most_common[0][0]从而获取对应的类别标签

```
def knn(X_train, y_train, X_test, k=3):
    y_pred = []

    for test_point in X_test:
        distances = []

        # 计算测试点与每个训练点之间的距离
        for x_train in X_train:
            distances.append(euclidean_distance(test_point, x_train))

        # 获取k个最近邻居的索引
        k_neighbors_indices = np.argsort(distances)[:k]

        # 获取k个最近邻居的类别
        k_neighbors_labels = [y_train[i] for i in k_neighbors_indices]

        # 确定最常见的类别作为预测类别
        most_common = Counter(k_neighbors_labels).most_common(1)
        y_pred.append(most_common[0][0])

    return y_pred
```

最后定义主方法读取数据集，之后进行划分，最后进行预测输出

```
if __name__ == '__main__':
    iris = pd.read_csv('iris.csv')

    data = iris.iloc[:, :4]
    target = iris.iloc[:, -1].values

    X_train, X_test, y_train, y_test = train_test_split(data.values, target,
                                                        test_size=0.2, random_state=41)

    predictions = knn(X_train, y_train, X_test, k=3)

    accuracy = np.mean(predictions == y_test)
    print(f"Accuracy: {accuracy:.2f}")
```

四、实验结果

运行结果如下图所示：

```
D:\anaconda3\python.exe D:\Desktop\learning\3
Accuracy: 0.93
```

接下来对 K 进行讨论：

1、当 K 的取值过小时，一旦有噪声成分存在们将会对预测产生比较大影响，例如 K 为 1 时，一旦最近的是噪声，那就会出现偏差，K 值的减小就意味着整体模型变得复杂，容易发生欠拟合。

2、如果 K 的值取的过大时，就相当于用较大邻域中的训练实例进行预测，学习的近似误差会增大。这时与输入目标点较远实例也会对预测起作用，使预测发生错误。K 值的增大就意味着整体的模型变得简单。

通过可视化的形式输出 K 与准确率的关系：

```python
k_range = range(1, 31)
k_scores = []

for k in k_range:
    predictions = knn(X_train, y_train, X_test, k)
    k_scores.append(np.mean(predictions == y_test))

plt.plot(k_range, k_scores)
plt.xlabel('Value of K for KNN')
plt.ylabel('Cross-Validated Accuracy')
plt.show()
```

运行结果如下所示，可以发现 K 过大、过小准确率都会降低，此外也可以发现奇数的 K 会比偶数更好一些，这样保证在计算结果最后会产生一个较多的类别，如果取偶数可能会产生相等的情况，不利于预测