厦門大學


# 软件学院

## 《人工智能导论》实验报告

题　　目 KNN 算法

姓　　名 陈澄

学　　号 32420212202930

班　　级 软工三班

实验时间 2024/05/09

**2024** 年 **05** 月 **09** 日

# 1 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。本实验通过解决 iris 数据集分类，帮助学生更好的熟悉和掌握 KNN 算法。

# 2 实验内容

使用 iris 数据集（可网上自行下载）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（这三个名词都是花的品种）。

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）。

# 3 实验步骤

1. 定义 IrisData

```cpp
struct IrisData {
    double sepalLength;
    double sepalWidth;
    double petalLength;
    double petalWidth;
    string species;
};
```

2. 计算欧式距离

```cpp
// 计算两个样本之间的欧氏距离
double euclideanDistance(const IrisData& a, const IrisData& b) {
    return sqrt(pow(a.sepalLength - b.sepalLength, 2) +
        pow(a.sepalWidth - b.sepalWidth, 2) +
        pow(a.petalLength - b.petalLength, 2) +
        pow(a.petalWidth - b.petalWidth, 2));
}
```

3. KNN 算法

```cpp
// KNN算法
string knn(const vector<IrisData>& trainData, const IrisData& testData, int k) {
    // 计算测试样本与训练样本的距离，并按距离升序排序
    vector<pair<double, string>> distances;
    for (const auto& trainSample : trainData) {
        double dist = euclideanDistance(trainSample, testData);
        distances.push_back(make_pair(dist, trainSample.species));
    }
    sort(distances.begin(), distances.end());

    // 统计前k个最近邻样本的类别
    unordered_map<string, int> freq;
    for (int i = 0; i < k; ++i) {
        freq[distances[i].second]++;
    }

    // 找到频率最高的类别
    int maxFreq = 0;
    string predClass;
    for (const auto& pair : freq) {
        if (pair.second > maxFreq) {
            maxFreq = pair.second;
            predClass = pair.first;
        }
    }

    return predClass;
}
```

4. Main()方法

读取数据集文件，调用 KNN 算法预测并分析正确率

```cpp
int main() {
    ifstream trainFile("D:\\learn\\大三下\\人工智能导论\\实验五决策树算法\\traindata.txt");
    ifstream testFile("D:\\learn\\大三下\\人工智能导论\\实验五决策树算法\\testdata.txt");
    if (!trainFile.is_open() || !testFile.is_open()) {
        cout << "Failed to open files." << endl;
        return 1;
    }
    // 示例数据集
    vector<IrisData> irisData, targetData;

    string line;
    while (getline(trainFile, line)) {
        stringstream ss(line);
        vector<double> values;
        double val;
        while (ss >> val) {
            values.push_back(val);
        }
        irisData.push_back({ values[0],values[1],values[2],values[3],to_string(values[4]) });
    }

    while (getline(testFile, line)) {
        stringstream ss(line);
        vector<double> values;
        double val;
        while (ss >> val) {
            values.push_back(val);
        }
        targetData.push_back({ values[0],values[1],values[2],values[3],to_string(values[4]) });
    }

    // 设置K值
    int k = 3;

    int error = 0;

    for (int i = 0; i < targetData.size(); i++) {
        string predClass = knn(irisData, targetData[i], k);
        if (predClass != targetData[i].species)error++;
    }


    cout << "Test Accuracy: " << (targetData.size() - error) * 100.0 / targetData.size() << "%" <<

    return 0;
}
```

## 4 运行结果

## 5 我的体会

在本次实验中，我深入研究了 K 最近邻（KNN）算法，并将其应用于分类问题。通过编程实践，我加深了对 KNN 算法的理解，包括其原理、优缺点以及在实际问题中的应用。我发现 KNN 算法简单直观，对非线性数据适用，并且可以处理多分类问题。然而，该算法在大型数据集上效率较低，对噪声和无关特征敏感。在实验中，我注意到调优参数如 K 值和距离度量方法对算法性能的影响。总的来说，本次实验使我对 KNN 算法有了更深入的了解，并为未来解决分类问题提供了有益的经验。