# 分类算法

目录

[分类算法 1](#_Toc85142934)

[**SVM** 1](#_Toc85142935)

[**简述** 1](#_Toc85142936)

[**原理方法** 1](#_Toc85142937)

[**实现** 1](#_Toc85142938)

[**决策树** 2](#_Toc85142939)

[**简述** 2](#_Toc85142940)

[**原理方法** 2](#_Toc85142937)

[**实现** 3](#_Toc85142941)

[**BPNN** 4](#_Toc85142942)

[**步骤** 5](#_Toc85142943)

[**原理方法** 5](#_Toc85142937)

[**实现** 5](#_Toc85142944)

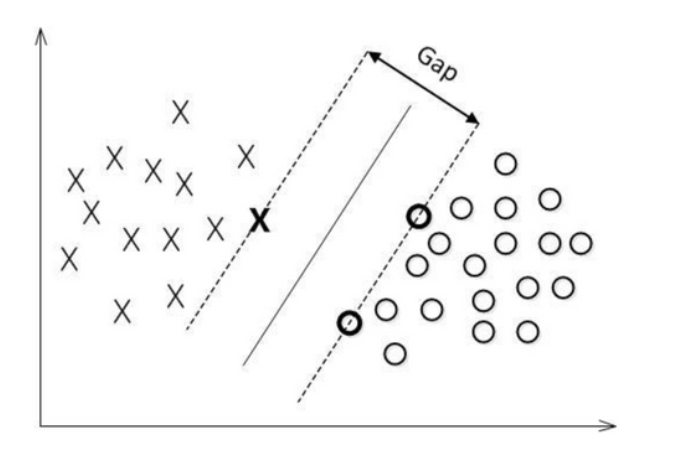
[**KNN** 5](#_Toc85142942)

[**概述及实现** 6](#_Toc85142944)

**SVM**

**简述**

可以把N个数据点看作是N维向量。用N-1维的超平面对这N个向量进行分隔开来。而SVM便是选择超平面，使其到离它最近的数据点的距离最大。（具有最强的泛化力）



**原理方法**

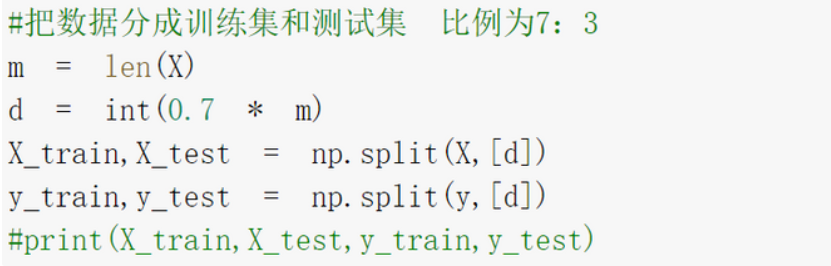
* 构造距离公式,确定和的值，使得点到该直线的距离最大
* 将公式转化为基本型，对目标函数进行凸二次规划。利用对偶问题，做到高效求解
* 由于拉格朗日具有对偶性，可以使用拉格朗日乘子法，将约束条件融合进目标函数求解
* 向量机分为三类：线性可分SVM、线性SVM、非线性SVM。对于非线性的问题，可以通过核方法将其转化为线性可分求解

**实现**

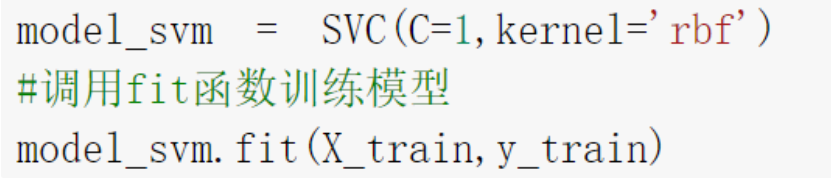
* 对数据进行导入，并进行Z-score标准化处理



* 把数据分按7：3为训练集和测试集



* 创建SVM模型，并调用fit函数对模型进行训练

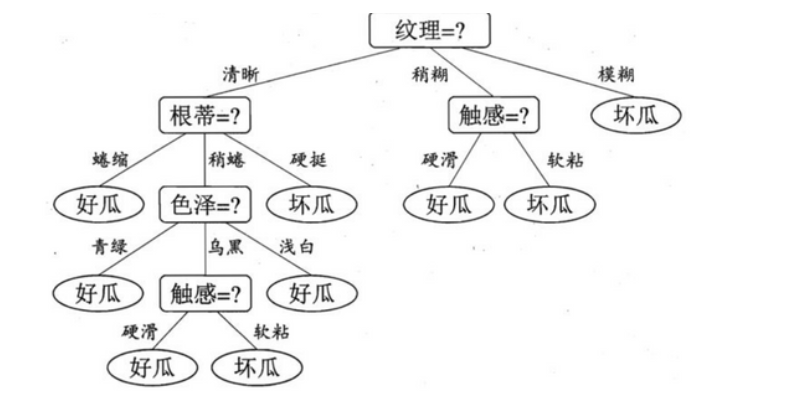


***SVC****：C是惩罚参数，C越大，趋向于训练集全对的情况，泛化能力越弱。kernel中的函数便是上面介绍的高斯核函数*

**决策树**

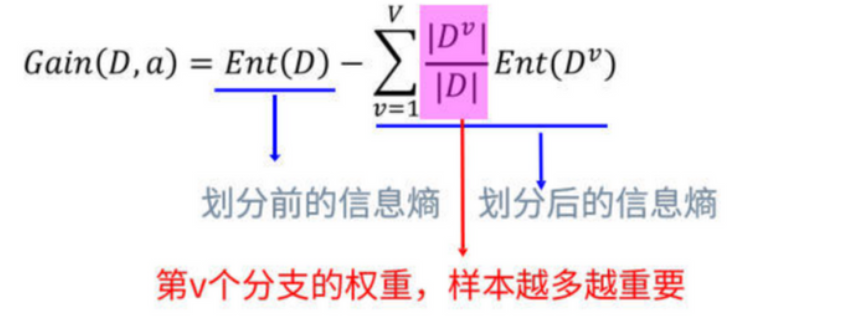
**简述**

顾名思义，决策树是一种树形结构。每个结点都作为一种判断条件，从而对输入进行分类。



**原理方法**

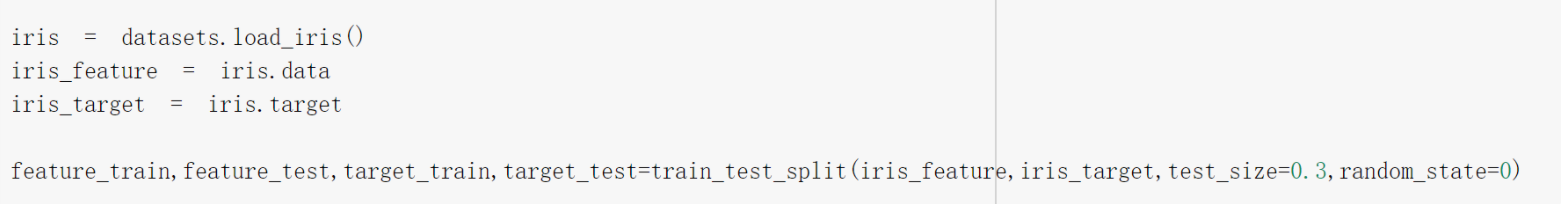
* 结点的分裂：如果一个结点无法给出判断。就将该结点分为两个子节点
* 结点的选择：目的是构建一个稳定有序的系统，就需要该系统的信息熵越低。常用的算法有ID3、C4.5和CART
* ID3：通过增益信息来构建决策树。使条件熵—信息熵的结果越小，该树构建的就越简单



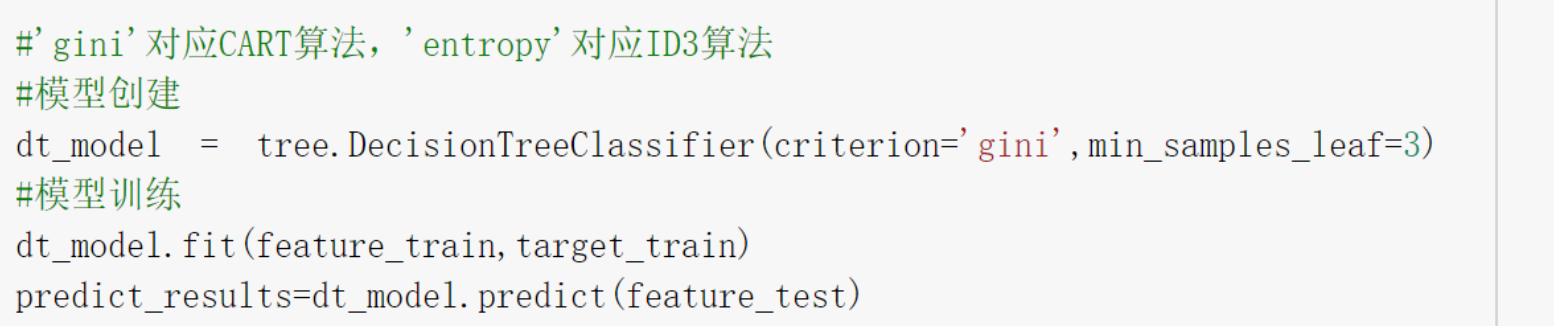
* C4.5：对ID3的改进，用信息增益率来选择最优的结点，选取值多的属性。同时克服了ID3不能处理连续属性的缺点
* CART：使用基尼系数来作为选择的标准

**实现**

* 数据的导入及化分测试集和训练集，比例分为7：3

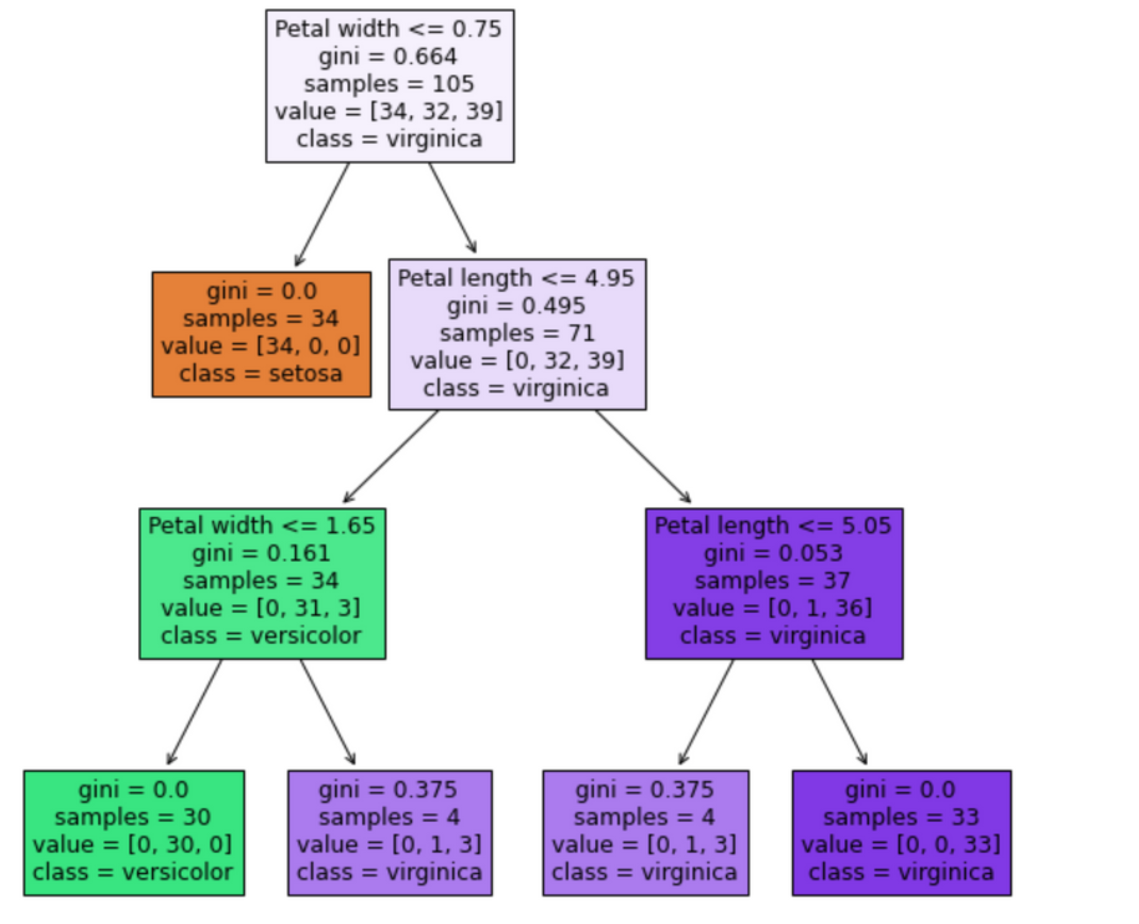


* 建立决策树模型（这里使用CART算法），对测试集进行训练



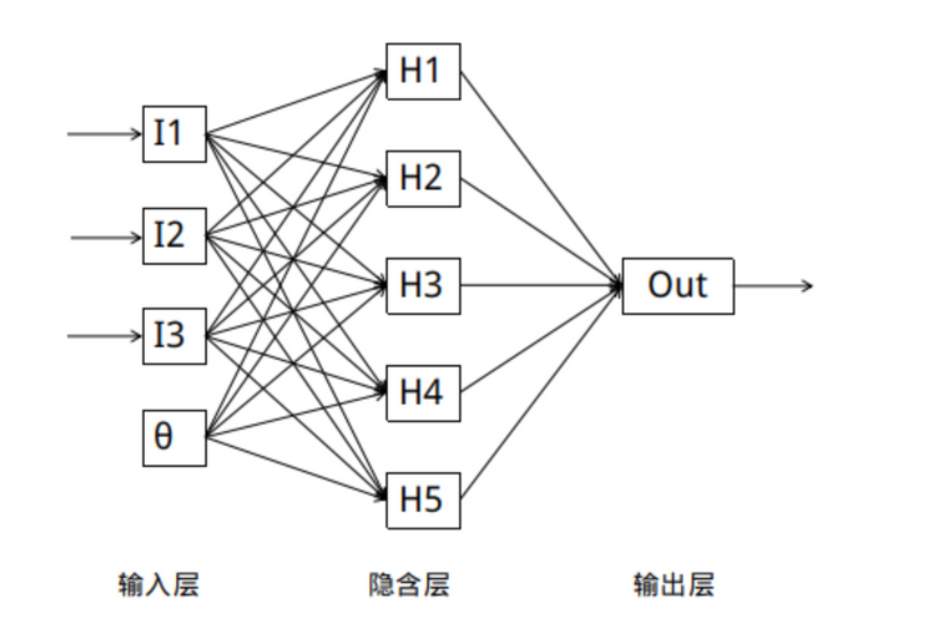
**DecisionTreeClassifier**

*criterion中设置算法类型，“gini”是使用的CART算法，”entropy“使用的是ID3算法*



**BPNN**

神经网络由输入层，隐含层和输出层构成。每个神经元表示对数据的一次处理。BP神经网络的训练过程即是根据前馈得到的预测值和参考值比较，根据误差调整权重的过程。

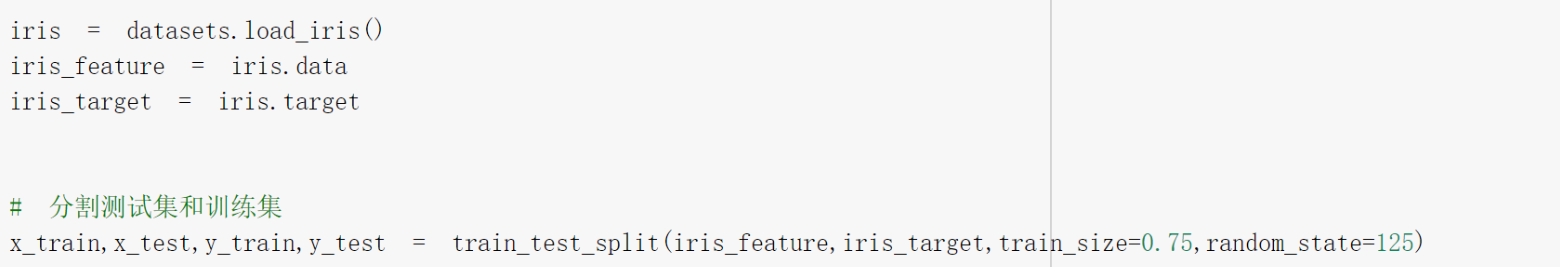


**原理方法**

* 定义sigmoid等工具函数
* 定义神经网络正向传播函数，并返回输出结果。首先先随机定义权值参数，隐含层接收来自输入层的数据，输出层接收来自隐含层的数据。
* 定义反向传播即回推的过程。运用梯度下降法，找到最小的权值并进行反馈修改
* 定义训练方法，设置最大学习次数和学习率
* 编写函数进行测试

**实现**

* 如上，设置训练集和测试集为8：2



* 调用BPNN函数



**solver**中的参数表示采用随机梯度下降的方法，**alpha**表示惩罚系数，**max\_iter**是最大迭代次数。但这里要注意，训练次数不能一味的大，会出现过拟合的情况。

**KNN**

**概述与实践**

该项目是我之前基于C++实现的。将数据集分为7：3，分别作为训练集和测试集。采取随机种子进行筛选数据。其次计算每个测试数据到训练数据的距离（分别采用了曼哈顿距离和欧拉距离两种方式）。选出距离最小的前K组数据，找出出现最多的鸢尾花种类，便认定该种类为测试集的种类，最后进行准确率测试。（估计是随机种子选取的问题，导致训练集中出现了重复的数据，影响的数据的准确率。最后的结果大概是0.7左右）

下面是部分代码

****

