Python机器学习基础教程

Stephen CUI[1]

February 17, 2023

[1]cuixuanStephen@gmail.com

Chapter 1

监督学习

1.1 分类与回归

监督机器学习问题主要有两种，分别叫作分类（classification）与回归（regression）。

分类问题的目标是预测类别标签（class label），这些标签来自预定义的可选列表。分类问题有时可分为二分类（binary classification，在两个类别之间进行区分的一种特殊情况）和多分类（multiclass classification，在两个以上的类别之间进行区分）。在二分类问题中，我们通常将其中一个类别称为正类（positive class），另一个类别称为反类（negative class）。这里的“正”并不代表好的方面或正数，而是代表研究对象。

回归任务的目标是预测一个连续值，编程术语叫作浮点数（floating-point number），数学术语叫作实数（real number）。

区分分类任务和回归任务有一个简单方法，就是问一个问题：输出是否具有某种连续性。如果在可能的结果之间具有连续性，那么它就是一个回归问题。

1.2 泛化、过拟合与欠拟合

在监督学习中，我们想要在训练数据上构建模型，然后能够对没见过的新数据（这些新数据与训练集具有相同的特性）做出准确预测。如果一个模型能够对没见过的数据做出准确预测，我们就说它能够从训练集**泛化**（generalize）到测试集。我们想要构建一个泛化精度尽可能高的模型。

判断一个算法在新数据上表现好坏的唯一度量，就是在测试集上的评估。然而从直觉上看[1]，我们认为简单的模型对新数据的泛化能力更好。因此，我们总想找到最简单的模型。构建一个对现有信息量来说过于复杂的模型，这被称为**过拟合**（overfitting）。如果你在拟合模型时过分关注训练集的细节，得到了一个在训练集上表现很好、但不能泛化到新数据上的模型，那么就存在过拟合。与之相反，如果你的模型过于简单，那么你可能无法抓住数据的全部内容以及数据中的变化，你的模型甚至在训练集上的表现就很差。选择过于简单的模型被称为**欠拟合**（underfitting）。

[1]在数学上也可以证明这一点。

Table 1.1: 一些样本数据集

名称	来源	特征量	用途
forge	模拟	26×2	二分类数据集
wave	模拟	40×1	回归算法数据集
cancer	真实	569×30	二分类数据集
boston	真实	506×13	回归数据集

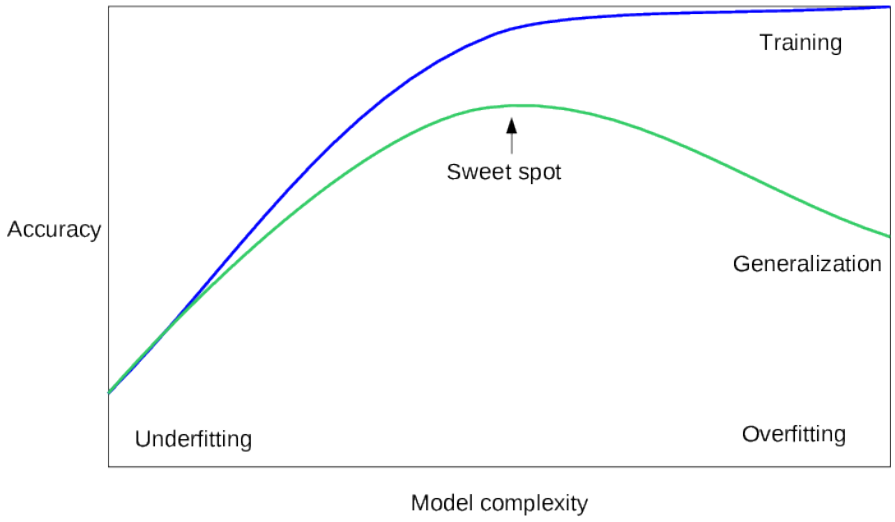

Figure 1.1: Trade-off of model complexity against training and test accuracy

模型复杂度与数据集大小的关系

需要注意，模型复杂度与训练数据集中输入的变化密切相关：数据集中包含的数据点的变化范围越大，在不发生过拟合的前提下你可以使用的模型就越复杂。通常来说，收集更多的数据点可以有更大的变化范围，所以更大的数据集可以用来构建更复杂的模型。但是，仅复制相同的数据点或收集非常相似的数据是无济于事的。

收集更多数据，适当构建更复杂的模型，对监督学习任务往往特别有用。本书主要关注固定大小的数据集。在现实世界中，你往往能够决定收集多少数据，这可能比模型调参更为有效。永远不要低估更多数据的力量！

1.3 监督学习算法

现在开始介绍最常用的机器学习算法，并解释这些算法如何从数据中学习以及如何预测。我们还会讨论每个模型的复杂度如何变化，并概述每个算法如何构建模型。我们将说明每个算法的优点和缺点，以及它们最适应用于哪类数据。此外还会解释最重要的参数和选项的含义，许多算法都有分类和回归两种形式。

1.3.1 一些样本数据集

我们将使用一些数据集来说明不同的算法。其中一些数据集很小，而且是模拟的，其目的是强调算法的某个特定方面。其他数据集都是现实世界的大型数据集。

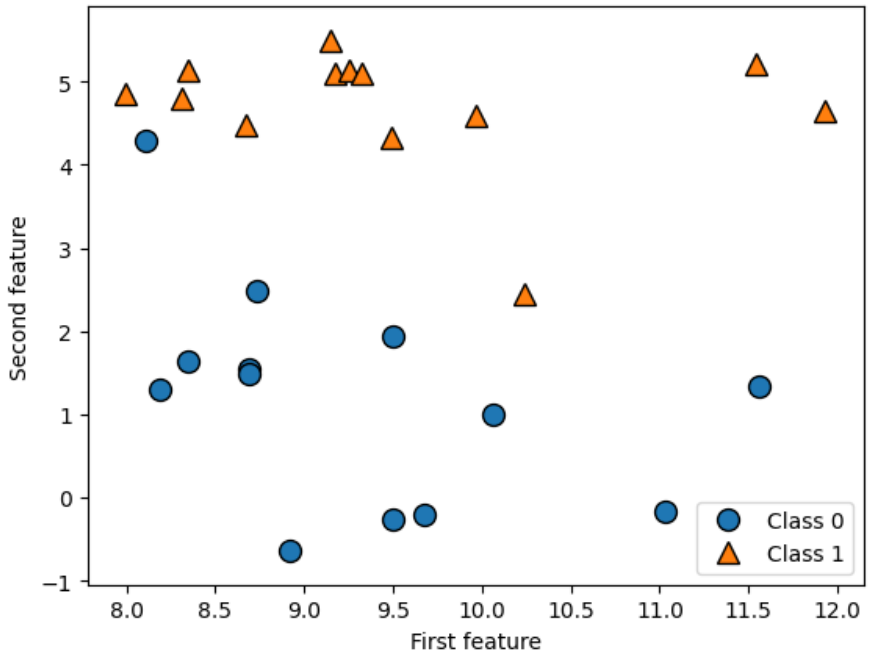

Figure 1.2: Scatter plot of the forge dataset

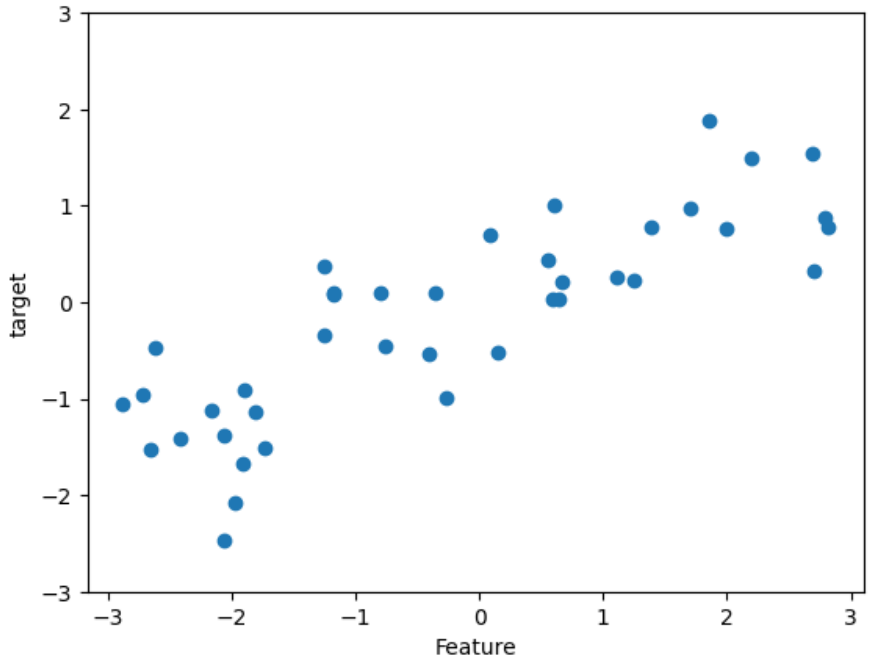

Figure 1.3: Plot of the wave dataset

从特征较少的数据集（也叫低维数据集）中得出的结论可能并不适用于特征较多的数据集（也叫高维数据集）。只要你记住这一点，那么在低维数据集上研究算法也是很有启发的。

包含在 scikit-learn 中的数据集通常被保存为 Bunch 对象，里面包含真实数据以及一些数据集信息。关于 Bunch 对象，你只需要知道它与字典很相似，而且还有一个额外的好处，就是你可以用点操作符来访问对象的值（比如用 `bunch.key` 来代替 `bunch['key']`）。

对于我们的目的而言，我们需要扩展boston数据集，输入特征不仅包括这 13 个测量结果，还包括这些特征之间的乘积（也叫**交互项**）。换句话说，我们不仅将犯罪率和公路可达性作为特征，还将犯罪率和公路可达性的乘积作为特征。像这样包含导出特征的方法叫作**特征工程**（feature engineering）。

1.3.2 k近邻

k-NN 算法可以说是最简单的机器学习算法。构建模型只需要保存训练数据集即可。想要对新数据点做出预测，算法会在训练数据集中找到最近的数据点，也就是它的“最近邻”。

k近邻分类 k-NN 算法最简单的版本只考虑一个最近邻，也就是与我们想要预测的数据点最近的训练数据点。预测结果就是这个训练数据点的已知输出。Figure 1.4给出了这种分类方法在 forge 数据集上的应用。

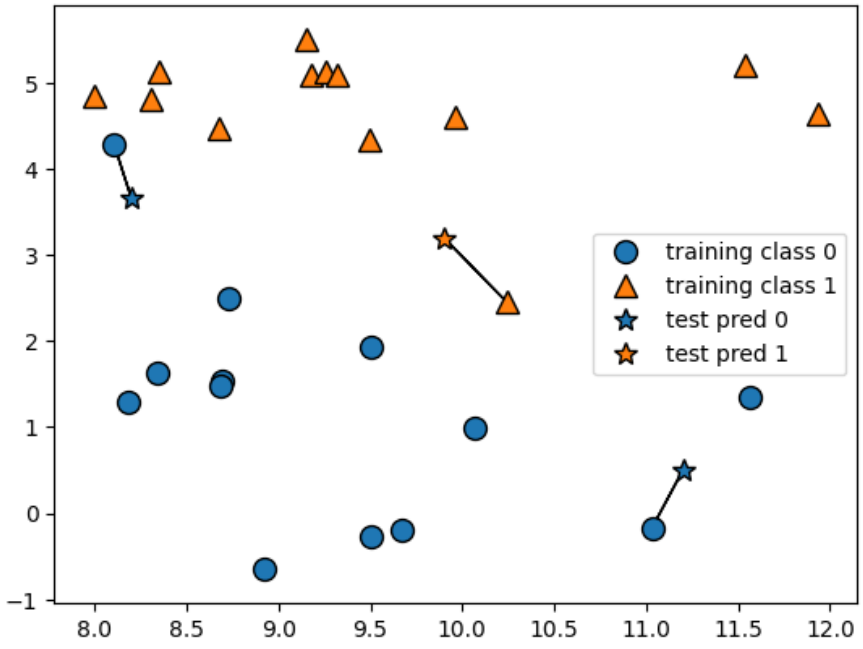

Figure 1.4: Predictions made by the one-nearest-neighbor model on the forge dataset

除了仅考虑最近邻，我还可以考虑任意个（k 个）邻居。这也是 k 近邻算法名字的来历。在考虑多于一个邻居的情况时，我们用“投票法”（voting）来指定标签。将出现次数更多的类别（也就是 k 个近邻中占多数的类别）作为预测结果。

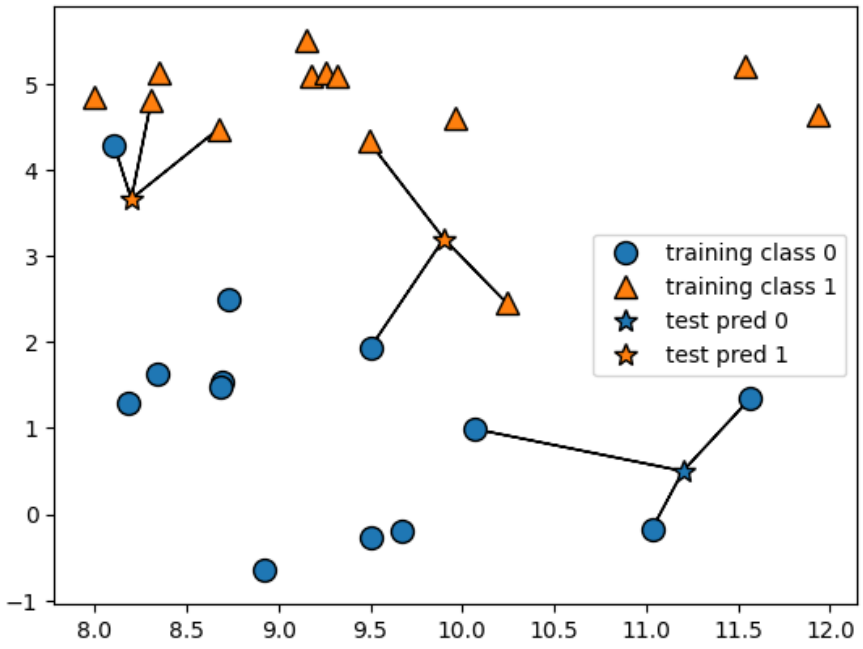

Figure 1.5: Predictions made by the three-nearest-neighbors model on the forge dataset

分析KNeighborsClassifier 分别将 1 个、3 个和 9 个邻居三种情况的决策边界可视化，见Figure 1.6。

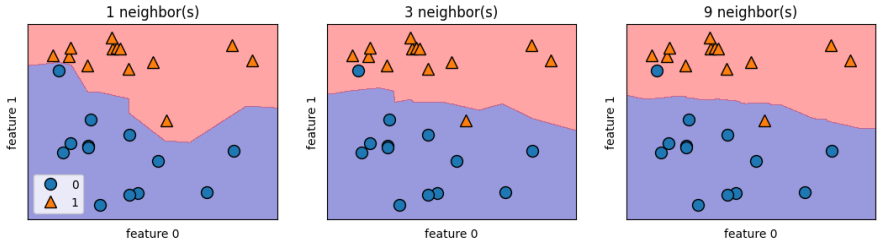

Figure 1.6: Decision boundaries created by the nearest neighbors model

从左图可以看出，使用单一邻居绘制的决策边界紧跟着训练数据。随着邻居个数越来越多，决策边界也越来越平滑。更平滑的边界对应更简单的模型。换句话说，使用更少的邻居对应更高的模型复杂度（如Figure 1.1右侧所示），而使用更多的邻居对应更低的模型复杂度（如Figure 1.1左侧所示）。假如考虑极端情况，即邻居个数等于训练集中所有数据点的个数，那么每个测试点的邻居都完全相同（即所有训练点），所有预测结果也完全相同（即训练集中出现次数最多的类别）。