COMP 551 Lecture 2 - K-Nearest Neighbours (M2)

Junji Duan

2024/1/10

Today's Outline

- Learning objectives
- Nearest Neighbour
- KNN with K ≥ 1
- The choice of K as a hyperparameter in KNN
- Time complexity
- Weighted KNN
- KNN regression

Terminologies of nearest neighbour (KNN) classification

Training

KNN 算法在训练阶段实际上并不进行任何“学习”处理, 它只是“记住”了训练数据集。这意味着 KNN 直接存储了所有的训练数据, 用于后续的预测任务。

- 训练集 D：包含所有训练样本的集合, 表示为 $D = \left\{\left(x^{(n)}, y^{(n)}\right)\right\}_{n=1}^{N}$。
- $x^{(n)}$：第 n 个训练样本的 D-维输入特征。
- $y^{(n)}$：第 n 个训练样本的分类输出 (标签)。
- N：训练样本的数量, 即所有 $\left(x^{(n)}, y^{(n)}\right)$ 对的总数。
- n：训练样本的 INDEX 索引。

由于 KNN 直接依赖于实例数据进行学习和预测, 因此它被称为基于实例的学习器（examplar-based learner）或非参数方法（non-parametric method）。

Prediction

KNN 预测新数据点的标签是基于训练集中最相似的 K 个样本的标签来进行的。具体来说:

- 当需要对一个新的数据点进行分类时, KNN 算法会计算该点与训练集中每个样本之间的距离。
- 然后选取距离最近的 K 个样本。
- 这 K 个最近邻样本中最频繁出现的类别将被选为新数据点的预测类别。

KNN 的核心思想是相似的样本应该具有相似的输出。该算法的性能很大程度上取决于 K 的选择和距离度量的方法（如欧氏距离、曼哈顿距离等）。KNN 简单直观，但如果训练数据集非常大, 则在计算和存储上可能非常昂贵。此外, 它对噪声数据和非相关特征 [irrelevant features] 也比较敏感。

Pairwise Distance

在 K 最近邻（K-Nearest Neighbor，简称 KNN 算法中, 预测新数据点 $\mathbf{x}^{**}$ 的关键步骤是在训练集中找到 K 个最相似的样本。首先, 我们考虑 $K = 1$, 即我们想找到训练集中与新数据点 $\mathbf{x}^{(*)}$ 最相似的一个样本。

相似性度量 (Similarity Measure)
在 KNN 算法中, 有多种方式可以衡量数据点之间的相似性。一种常见的方法是通过欧几里得距离 (Euclidean Distance) 来计算:

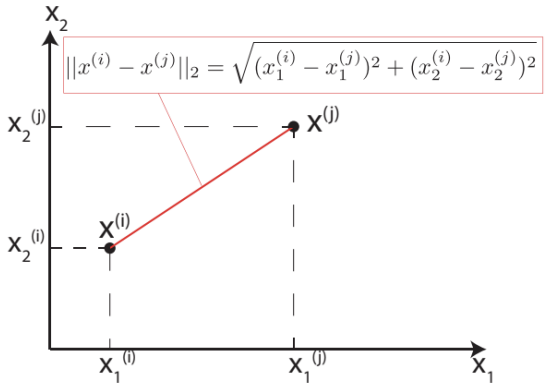
$$\text{distance}\left(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}\right) = \left\|\mathbf{x}^{(i)} - \mathbf{x}^{(j)}\right\|_2 = \sqrt{\sum_{d=1}^{D}\left(x_d^{(i)} - x_d^{(j)}\right)^2}$$

这里, 欧几里得距离是指两个数据点之间连线的长度, 这种距离在直观上表示了两点在空间中的“直线”距离。

二维示例 (Two-dimensional Example)
在二维空间中, 如果我们有两个点 $\mathbf{x}^{(i)}$ 和 $\mathbf{x}^{(j)}$, 则它们之间的欧几里得距离可以通过上述公式直接计算出来, 该公式基本上是应用了勾股定理来求直角三角形的斜边长。

通过这种方式, KNN 算法能够评估新数据点与每个训练数据点之间的距离, 进而确定最近的邻居。选择距离最小的邻居作为新数据点的预测类别（在 $K = 1$ 的情况下），或者选择 K 个最近的邻居, 并基于它们的类别通过多数投票等方法来确定新数据点的类别。

这种基于距离的方法使得 KNN 在处理分类问题时直观且易于实现，但同时也受到了计算成本和维度灾难的限制, 尤其是在高维数据中。



对于一个新的数据点 $\mathbf{x}^{(*)}$, KNN 算法通过以下步骤进行预测:

1. 计算距离: 首先, 计算新数据点 $\mathbf{x}^{(*)}$ 与训练集中每一个数据点 $\mathbf{x}^{(i)}$ 之间的距离。常见的距离度量方法包括:

- 欧几里得距离（Euclidean Distance）:

$$\sqrt{\sum_{d=1}^{D}\left(x_d^{(i)} - x_d^{(j)}\right)^2}$$

- 参数:
  - $x_d^{(i)}$: 第 i 个数据点的第 d 个特征值。
  - $x_d^{(j)}$: 第 j 个数据点的第 d 个特征值。
  - D : 特征的总数, 即数据点的维度。
- 描述: 欧几里得距离测量两个点在多维空间中的直线距离, 适用于连续特征。

- 曼哈顿距离（Manhattan Distance）:

$$\sum_{d=1}^{D}\left|x_d^{(i)} - x_d^{(j)}\right|$$

- 参数:
  - $x_d^{(i)}$ 和 $x_d^{(j)}$: 与欧几里得距离相同, 代表两个比较点的对应特征值。
  - D : 特征总数。
- 描述: 曼哈顿距离是通过计算两点在各个维度的绝对差值的总和来衡量的, 适合城市街区距离 (L1 范数), 对异常值具有较高的鲁棒性。

- 余弦相似度（Cosine Similarity）(通常用于计算相似度, 而不是距离):

$$\frac{\sum_{d=1}^{D} x_d^{(i)} x_d^{(j)}}{\left(\sqrt{\sum_d \left(x_d^{(i)}\right)^2}\sqrt{\sum_d \left(x_d^{(j)}\right)^2}\right)}$$

- 参数:
  - $x_d^{(i)}$ 和 $x_d^{(j)}$: 与欧几里得距离相同, 代表两个比较点的对应特征值。
  - D : 特征总数。
- 描述: 余弦相似度不是直接测量距离, 而是测量两个向量在方向上的差异。值范围从 - 1 (完全相反) 到 1 (完全相同), 常用于文本分析中的文档或向量相似度计算。

- 汉明距离（Hamming Distance）(适用于离散或分类输入特征):

$$\sum_{d=1}^{D} \mathbb{I}\left(x_d^{(i)} \neq x_d^{(j)}\right)$$

- 参数:
  - $x_d^{(i)}$ 和 $x_d^{(j)}$: 代表两个比较点在同一特征上的值, 通常是离散或分类特征。
  - D : 特征总数。
- 描述: 汉明距离是两个字符串或向量在相同位置上不同字符的数量。适用于分类特征的比较, 例如错误检测和纠正算法中。

---

2. 选择最近的邻居: 找到与 $\mathbf{x}^{(*)}$ 距离最近的训练数据点 $\mathbf{x}^{(i)}$, 即最相似的例子。
3. 预测标签: 将找到的最近邻居的标签 $y^{(i)}$ 乍为新数据点 $\mathbf{x}^{(*)}$ 的预测标签。

分析与思考

- $K = 1$ 的灵活性: 当 $K = 1$ 时, 模型可能过于灵活 (flexible), 容易受到噪声的影响, 从而导致过拟合。决策边界可能会过于复杂和不规则。

- $K = N$ 的情况: 如果 K 等于训练样本总数 N, 那么模型将非常僵化 (rigid), 可能只能预测出训练数据中最常见的类别, 忽略了数据的细微差别。

通过这种方式，KNN 算法能够根据数据点之间的相似性进行有效的分类，但它对参数 K 的选择非常敏感, 这需要根据具体的数据和问题进行调整。

离散类别预测

当使用 KNN 进行离散类别预测时, 新数据点 $\mathbf{x}^{(*)}$ 的预测标签是其 K 个最近邻居中最频繁出现的类别:

$$\hat{y}^{(*)} = \arg\max_{c} \sum_{n \in \mathcal{N}_K\left(\mathbf{x}^{(*)}, \mathcal{D}\right)} \mathbb{I}\left(y^{(n)} = c\right)$$

其中:

- $\arg\max$ 表示 “使得后面的表达式达到最大值的参数”，这里的参数是类别标签 c 。
- c 是类别的标识, 如 $\{1, 2, \ldots, C\}$ 中的任一类别。
- 这个表达式后面通常会跟一个求和表达式, 该求和表达式计算每个类别标签在最近的 K 个邻居中出现的次数。
- $\mathcal{N}_K\left(\mathbf{x}^{(*)}, \mathcal{D}\right)$ 表示基于某种距离函数（例如欧几里得距离）选出的对于新数据点 $\mathbf{x}^{(*)}$ 来说最接近的 K 个样本。
- $\mathcal{D} = \left\{\mathbf{x}^{(n)}, y^{(n)}\right\}_{n=1}^{N}$ 是训练数据集。
- 当 $K = 1$ 时, KNN 简化为寻找最相似的单一样本。
- $\mathbb{I}\left(y^{(n)} = c\right)$ 是一个指示函数, 当样本 n 的类别 $y^{(n)}$ 等于 c 时返回 1 , 否则返回 0 。

类别概率预测

在 KNN 中计算每个类别的概率如下:

$$p\left(y^{(*)} = c \mid \mathbf{x}^{(*)}\right) = \frac{1}{K} \sum_{n \in \mathcal{N}_K\left(\mathbf{x}^{(*)}, \mathcal{D}\right)} \mathbb{I}\left(y^{(n)} = c\right)$$

其中:

- $\mathcal{N}_K\left(\mathbf{x}^{(*)}, \mathcal{D}\right)$ 和训练数据集 $\mathcal{D}$ 的定义与离散类别预测相同。
- 当 $K = 1$ 时, 算法同样简化为找到最相似的单一样本。

举例说明 ($K = 3$)

假设对于新数据点 $\mathbf{x}^{(*)}$, 其最近的三个训练样本的索引为 n_1, n_2, n_3, 且这些样本的类别标签分别为 $y^{(n_1)} = 1, y^{(n_2)} = 1, y^{(n_3)} = 3$ 。根据 KNN 的类别概率预测公式，每个类别的概率为:

- 类别 1 的概率 $p\left(y^{(*)} = 1 \mid \mathbf{x}^{(*)}\right) = \frac{2}{3}$ (因为有两个邻居属于类别 1)
- 类别 2 的概率 $p\left(y^{(*)} = 2 \mid \mathbf{x}^{(*)}\right) = 0$ (因为没有邻居属于类别 2)
- 类别 3 的概率 $p\left(y^{(*)} = 3 \mid \mathbf{x}^{(*)}\right) = \frac{1}{3}$ (因为有一个邻居属于类别 3)

通过这种方式, KNN 不仅可以做出分类决策, 还可以给出新数据点属于各个类别的可能性, 为决策提供了额外的概率信息。

K 值的选择

K 值的选择是模型的一个超参数 (hyper-parameter)，这意味着它不是从数据中学习得来的, 而是需要在模型训练过程中通过经验设定的。不同的 K 值可能会对模型的预测准确率产生显著影响。

例如，在某个具体的情况下，当 $K = 1$ 时，准确率可能是 72%; 当 $K = 5$ 时, 准确率提高到 80% ；而 $K = 15$ 时，准确率又降到了 76% 。

如何选择 K

- 目标：目标是准确预测不在训练集中的未见数据。
- 准确率估计：通常通过测试集上的准确率来近似估计这一目标。
- 数据划分：为了有效选择 K, 需要将整个数据集划分为训练集、验证集和测试集。
- 使用训练集来 “训练” KNN 模型。
- 使用验证集来选择从一组有限的 K 值中表现最佳的那一个, 即选择在验证集上准确率最高的 K 。
- 最后, 使用测试集来评估选定的模型。

使用验证集选择 K (具体操作)

- 在训练数据中进一步划分, 例如将 100 个样本分为 50% 的训练和 50% 的验证。
- 在验证集上测试不同的 K 值, 记录每个 K 值的准确率。
- 选择在验证集上准确率最高的 K 值。例如, 如果测试发现 $K = 9$ 时在验证集上表现最佳, 则选择这个 K 值。

在实际操作中, 可以通过绘制 K 值与验证准确率的关系图来帮助决策。这种图通常会显示出一个或多个准确率峰值，表明这些 K 值可能是最偶 $\downarrow$ 泽。此外，选择 K 时还应考虑到模型的复杂性：较小的 K 可能导致模型对训练数据过度敏感（过拟合 Overfitting），而较大的 K 可能导致模型过于简化，忽略数据中的重要模式（欠拟合 Underfitting）。因此，通过验证集来平衡这一点是选择 K 的关键。

时间复杂度

- 欧几里得距离：我们假设使用欧几里得距离计算点之间的距离。对于训练集中的每个数据点, 计算距离的复杂度为 $O(D)$, 对于 N 个数据点的总复杂度为 $O(ND)$ 。
- 查找最小距离的 K 个点: 这个过程的复杂度为 $O(NK)$, 更精确地说是 $\sum_{k=0}^{K-1}(N-k)$, 即首先找到最小的, 然后是第二小的, 依此类推。
- 单个测试查询的计算复杂度：总复杂度为 $O(ND+NK)$ 。

改进最近邻搜索：使用 $k-d$ 树

- 基本思想：通过递归地聚类训练集来构建一棵树，然后通过这棵树搜索 K 个最近邻居，而不必查看所有 N 个训练点, 复杂度大约为 $O\left(2^D+\log N\right)$ 。

KNN 对特征缩放 (Feature Scaling) 的敏感性

- 问题描述: 例如, 将花萼宽度缩放 100 倍后, 可能因为花萼长度的影响变得微不足道, 导致预测失误, 因为在总距离中, 宽度成了主要因素。
- 解决方法: 在进行 KNN 预测之前, 对每个特征进行标准化处理：

$$x_d^{(n)}=\frac{x_d^{(n)}-\bar{x}_d}{sd\left(x_d\right)}$$

其中 $\bar{x}_d$ 是特征 d 的均值, $sd\left(x_d\right)$ 是特征 d 的标准差。

KNN 对随机无关特征的敏感性

- 问题描述：添加一个从标准正态分布采样的随机噪声特征, 并按比例缩放, 这可能会降低 KNN 的预测准确性，因为现有的 KNN 实现无法区分特征的重要性。
- 解决方法：移除与标签低相关的特征。

维数灾难 (Curse of Dimensionality)

- 基本概念：随着维度的增加, 空间的体积增长非常快, 可能需要在空间中搜索较远的点来找到最近邻。
- 影响：高维度下, 数据点分布稀疏, 大多数数据点对看起来彼此都很远, 这使得找到有意义的最近邻变得困难。

加权 KNN

- 定义：加权 KNN 根据每个样本与测试数据的距离来加权其贡献：

$$p\left(y^{(*)}=c \mid \mathbf{x}^{(*)}\right)=$$
$$\frac{1}{W_K\left(\mathbf{x}^{(*)}, \mathcal{D}\right)} \sum_{n \in \mathcal{N}_K\left(\mathbf{x}^{(*)}, \mathcal{D}\right)} \text{similarity}\left(\mathbf{x}^{(n)}, \mathbf{x}^{(*)}\right) \mathbb{I}\left(y^{(n)}=c\right)$$

其中, 相似度可以是欧几里得距离的倒数或余弦相似度。

KNN 回归

- 适用于回归：将分类的 KNN 方法应用于回归问题, 计算方式如下：

$$y^{(*)}=\frac{1}{K} \sum_{n \in \mathcal{N}_K\left(\mathbf{x}^{(*)}, \mathcal{D}\right)} y^{(n)}$$

或使用加权方法：

$$y^{(*)}=\frac{1}{W_K\left(\mathbf{x}^{(*)}, \mathcal{D}\right)} \sum_{n \in \mathcal{N}_K\left(\mathbf{x}^{(*)}, \mathcal{D}\right)} \text{similarity}\left(\mathbf{x}^{(n)}, \mathbf{x}^{(*)}\right) y^{(n)}$$

总结

- KNN 是一种非参数化的懒惰学习方法, 不需要在训练阶段做任何操作, 但测试时的时间复杂度和空间复杂度随训练数据的增大而增大。
- KNN 在有大量数据的情况下表现良好, 但对特征缩放和噪声敏感, 且受维数灾难 (Curse of Dimensionality) 的影响。