INT305 W1

Nearest Neighbor

数学表达 以及 回归和分类：

Mathematically, our training set consists of a collection of pairs of an input vector $\mathbf{x} \in \mathbb{R}^d$ and its corresponding target, or label, t

- Regression: t is a real number (e.g. stock price)
- Classification: t is an element of a discrete set $\{1, \dots, C\}$
- These days, t is often a highly structured object (e.g. image)

Denote the training set $\{(\mathbf{x}^{(1)}, t^{(1)}), \dots, (\mathbf{x}^{(N)}, t^{(N)})\}$

- Note: these superscripts have nothing to do with exponentiation!

Nearest Neighbor

- 问题：有一个 input vector x，要对它进行分类。
- 想法：在 training set 中，找到 **一个** 和 x 最像的（即：欧式距离最近的）vector x^* ，那么 x^* 的 label t^* 就可以看作 x 的 label y。
- 算法：

  欧几里得距离（Euclidean distance）：

  $$\left\|\mathbf{x}^{(a)} - \mathbf{x}^{(b)}\right\|_2 = \sqrt{\sum_{j=1}^{d} \left(x_j^{(a)} - x_j^{(b)}\right)^2}$$

  算法：

  1，Find example (x^*, t^*) (from the stored training set) closest to x. That is:

  $$\mathbf{x}^* = \underset{\mathbf{x}^{(i)} \in \text{ train. set}}{\operatorname{argmin}} \ \text{distance}\left(\mathbf{x}^{(i)}, \mathbf{x}\right)$$

  注：argmin（argument of the minimum）表示使目标函数取最小值时的变量值。

  2，Output y = t^*

k-Nearest Neighbors（KNN）

- 问题：training set 中会存在很多 noisy sample（或 mis-labeled data），这会影响结果的准确性。
- 解决方法：使用多个 sample 共同判断。

  前面的 Nearest Neighbor 只找到一个最近的 sample 作为依据进行判断，如果这个 sample 是 noise，那么就会出现错误。因此，找到 k 个最近的 samples 一起进行判断，就是 KNN。
- 算法：

  1，Find k examples {$x^{(i)}$, $t^{(i)}$} closest to the test instance x

  2，Classification output is majority class

$$y = \arg\max_{\mathbf{t}^{(z)} \in \mathbf{t}^{(i)}} \sum_{i=1}^{k} \mathbb{I}\left(t^{(z)} = t^{(i)}\right)$$

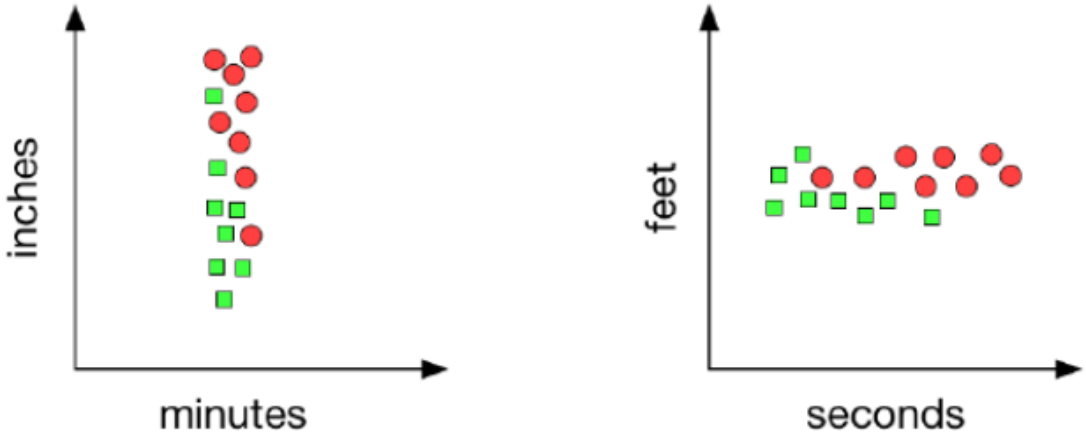
𝕀{statement} is the identity function and is equal to one whenever the statement is true. We could also write this as $\sigma(t^{(z)}, t^{(i)})$, with $\sigma(a, b) = 1$ if a = b, 0 otherwise.

注：第一步找到 k 个最近的 samples，用 $\{x^{(i)}, t^{(i)}\}$ 表示，其中 $x^{(i)}$ 和 $t^{(i)}$ 中都有 k 个值。

第二步根据 k 个 sample 的 label，找出数量最多的那个类，就是最后的输出。

- Tradeoffs in choosing k
  - small k
    - 擅长捕捉细颗粒度的特征（fine-grained patterns）
    - 可能会 overfit，即对 training data 中的随机特征敏感
  - large k
    - 可以通过对大量 sample 进行平均，做出稳定的预测
    - 可能 underfit，即无法捕捉某些重要的规律
  - balancing k
    - 最优 的 k 的值，取决于 data points n 的数量
    - 经验法则：choose k < $\sqrt{n}$
- Choosing k using validation set
  - k is an example of a **hyperparameter**
  - we can tune hyperparameters using a **validation set**

Curse of Dimensionality



- 问题：在计算距离时可能会出现以上的情况，绝对值大的feature 在欧式距离计算的时候起了决定性作用（在某一维度 或 feature 上紧密，在某一维度上分散）。
- 简单的解决方法：对每个维度的数据进行 **normalize**，使其变得 **零均值化**（zero mean，即使均值为 0；如图片像素值在 -128~128，均值为 0） 和 **单位方差化**（unit variance，即使方差为 1；方差是每个样本值与全体样本值的平均数之差的平方值的平均数，可以用来表示离散程度）。

  $$\tilde{x}_j = \frac{x_j - \mu_j}{\sigma_j}$$

  其中，x_j 为某个特征的原始值，μ_j 为该特征在所有样本中的平均值，σ_j 为该特征在所有样本中的标准差（标准差是方差的算术平方根），$\tilde{x}_j$ 为经过标准化处理后的特征值 **~ N(0, 1)**

Computational Cost

- number of computations at **training time**: 0（KNN 不需要 train）
- number of computations at **test time**, per query (naive algorithm)
    - calculuate D-dimensional Euclidean distance with N data points: ***O(ND)***（欧几里得距离要计算 D 个 features，一共 N 个点）
    - sort the distances: ***O(N logN)***