Nearest Neighbour Algorithms

Christos Dimitrakakis

September 19, 2024

Outline

Introduction

- The hidden secret of machine learning

The algorithm

- *k* Nearest Neighbours
- Extensions and parameters

Activities

Supervised learning

- Given labelled training examples $(x_1, y_1), \ldots (x_T, y_T)$ where
- $x_t \in \mathbb{R}^n$ are features
- $y_t \in Y$ are labels

Classification

- $Y = \{1, \ldots, m\}$ are discrete labels

Regression

- $Y = \mathbb{R}^m$ are continuous values

The kNN algorithm idea

- Assume an unknown example is similar to its neighbours
- Smoothness allows us to make predictions
- Lots of other algortihms

Performance of KNN on image classification

./fig/knn-image-performance.png

The Nearest Neighbour algorithm

Pseudocode

- Input: Data $(x_t, y_t)_{t=1}^{T}$, test point x, distance d
- $t^* = \arg\min_t d(x_t, x)$ / How do we implement this?
- Return $\hat{y}_t = y_{t^*}$

Classification

$\hat{y}_t \in [m] \equiv \{1, \ldots, m\}$

Regression

$\hat{y}_t \in \mathbb{R}^m$

The k-Nearest Neighbour algorithm

Pseudocode

- Input: Data $(x_t, y_t)_{t=1}^{T}$, test point x, distance d, neighbours k
- Calculate $h_t = d(x_t, x)$ for all t.
- Get sorted indices $s = \texttt{argsort}(h)$ so that $d(x_{s_i}, x) \leq d(x_{s_{i+1}}, x)$ for all i. (How?)
- Return $\sum_{i=1}^{k} y_{s_i}/k$.

Classification

- It is not convenient to work with discrete labels.
- We use a one-hot encoding (0, . . . , 0, 1, 0, . . . , 0)$.
- $y_t \in \{0, 1\}^m$ with $\|y_t\|_1 = 1$, so that the class of the t-th example is j iff $y_{t,j} = 1$.

Regression

- $y_t \in \mathbb{R}^m$, so we need do nothing

Making a decision

kNN Output

- Given features x, we get a vector
- $p_i = \hat{\mathbb{P}}(y = i|x)$.

Accuracy

- Predicted label a
- Actual label y
- $U(a, y) = \mathbb{I}\left\{a = y\right\}$

Classification decision to maximise accuracy

- $a_t = \arg\max_i p_i$

The number of neighbours

$k = 1$

- How does it perform on the training data?
- How might it perform on unseen data?

$k = T$

- How does it perform on the training data?
- How might it perform on unseen data?

Distance function

For data in $\mathbb{R}^n$, p-norm

$$d(x, y) = \|x - y\|_p$$

Scaled norms

When features having varying scales:

$$d(x, y) = \|Sx - Sy\|_p$$

Or pre-scale the data

Complex data

- Manifold distances
- Graph distance

Distances

A distance $d(\cdot, \cdot)$:

- Identity $d(x, x) = 0$.
- Positivity $d(x, y) > 0$ if $x \neq y$.
- Symmetry $d(y, x) = d(x, y)$.
- Triangle inequality $d(x, y) \leq d(x, z) + d(z, y)$.

For data in $\mathbb{R}^n$, p-norm

$$d(x, y) = \|x - y\|_p$$

Norms;

A norm $\|\cdot\|$

- Zero element $\|0\| = 0$.
- Homogeneity $\|cx\| = c\|x\|$ for any scalar a.
- Triangle inequality $\|x + y\| \leq \|x\| + \|y\|$.

\$p\$-norm

$$\|z\|_p = \left(\sum_i z_i^p\right)^{1/p}$$

Neighbourhood calculation

If we have T datapoints

Sort and top K.

- Requires $O(T \ln T)$ time

Use the Cover-Tree or KD-Tree algorithm

- Requires $O(cK \ln T)$ time.
- c depends on the data distribution.

Class data

Fill in the class data

Figure: Link to spreadsheet

KNN activity

- Implement nearest neighbours
- Introduction to scikitlearn nearest neighbours