Week 3. k-Nearest Neighbours (kNN)

Dr. Shuo Wang

Overview

- Intuitive understanding
- The kNN algorithm
- Pros/cons

Parametric and Non-parametric Models

Parametric models:

- A model that summarizes data with a finite set of parameters.
- Make assumptions on data distributions.
- E.g. linear/logistic regression, neural networks

Non-parametric models:

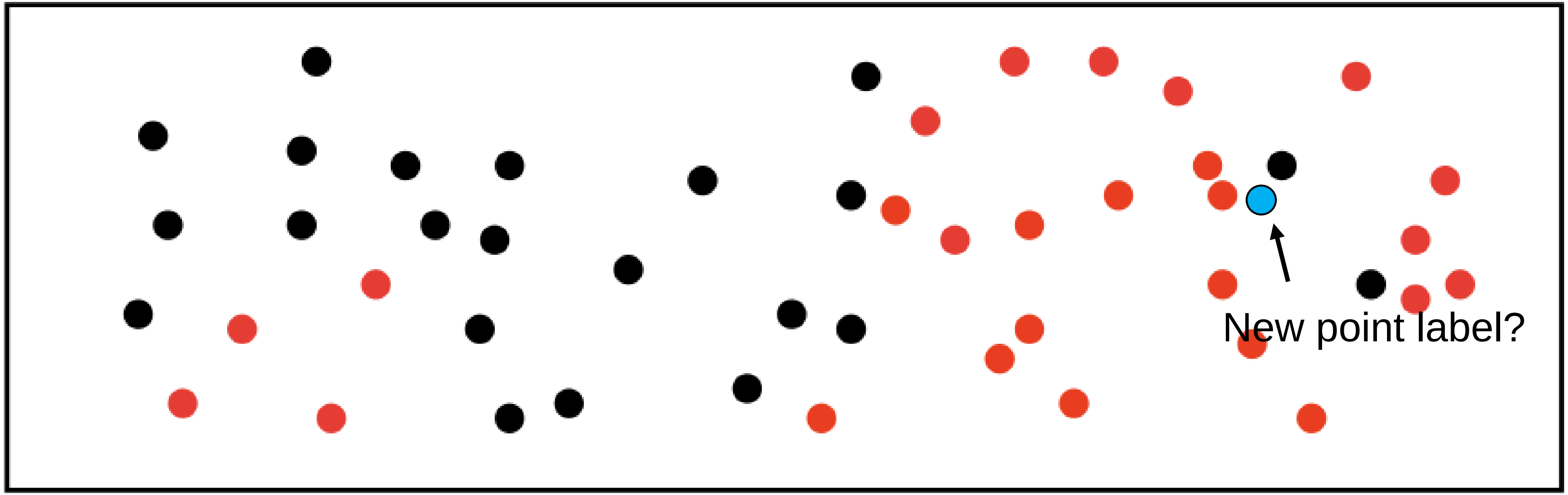
- A model that cannot be characterized by a bounded set of parameters.
- No assumptions on data distributions.
- E.g. instance-based learning that generate hypotheses using training examples, including kNN, SVM, decision trees, etc.

kNN Basics

- Full name: k-Nearest Neighbours (kNN, or k-NN).
- It is nonparametric.
  No assumption about the functional form of the model.
- It is instance-based.
  The prediction is based on a comparison of a new point with data points in the training set, rather than a model.
- It is a lazy algorithm.
  No explicit training step. Defers all the computation until prediction.
- Can be used for both classification and regression problems.

Intuitive Understanding

Instead of approximating a model function $f(x)$ globally, kNN approximates the label of a new point based on its nearest neighbours in training data.



Q1: How to choose k? e.g. let k = 3 to avoid issues.

Q2: how to we measure the distance between examples?

Distance metrics (or similarity metrics)

Given two points $\boldsymbol{x}^{(1)} = \left(x_1^{(1)}, x_2^{(1)}, \dots, x_d^{(1)}\right), \boldsymbol{x}^{(2)} = \left(x_1^{(2)}, x_2^{(2)}, \dots, x_d^{(2)}\right)$ in a d-dimensional space:

- Minkowski distance (or L^p norm)

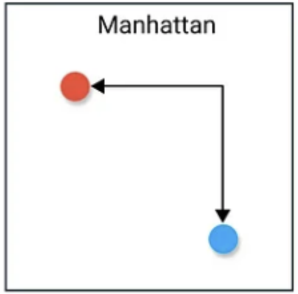
$$D\left(\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}\right) = \sqrt[p]{\sum_{i=1}^{d} \left|x_i^{(1)} - x_i^{(2)}\right|^p}$$

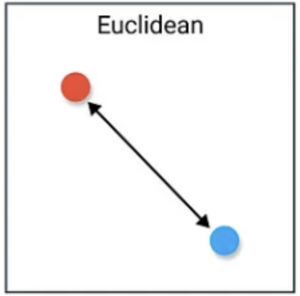
- When p=1, it becomes Manhattan distance

$$D\left(\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}\right) = \sum_{i=1}^{d} \left|x_i^{(1)} - x_i^{(2)}\right|$$

- When p=2, it becomes Euclidean distance

$$D\left(\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}\right) = \sqrt{\sum_{i=1}^{d} \left|x_i^{(1)} - x_i^{(2)}\right|^2}$$





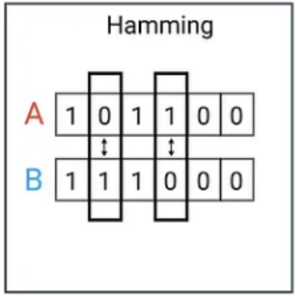
Distance metrics in kNN (common choice)

- Euclidean distance for real values (also called L^2 distance).

$$D\left(\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}\right) = \sqrt{\sum_{i=1}^{d} \left|x_i^{(1)} - x_i^{(2)}\right|^2}$$

- Hamming distance for discrete/categorical values, e.g. $x \in \{rainy, sunny\}$.

$$D\left(x^{(1)}, x^{(2)}\right) = \begin{cases} 0, if\ x^{(1)} = x^{(2)} \\ 1, \quad otherwise \end{cases}$$



Example

Consider a binary problem (lemon or orange) with 2 dimensions (height and width) with following training examples:

- $\boldsymbol{x}^{(1)}$ =(6,6), $y^{(1)}$ =orange
- $\boldsymbol{x}^{(2)}$ =(8,10), $y^{(2)}$ =lemon
- $\boldsymbol{x}^{(3)}$ =(7,6), $y^{(3)}$ =orange

New example

- $\boldsymbol{x}^{(4)}$ =(8,7), $y^{(4)}$ =? Using k=1 nearest neighbour
- $D\left(\boldsymbol{x}^{(4)}, \boldsymbol{x}^{(1)}\right) = \sqrt{\sum_{i=1}^{d} \left|x_i^{(4)} - x_i^{(1)}\right|^2} = \sqrt{(x_1^{(4)} - x_1^{(1)})^2 + (x_2^{(4)} - x_2^{(1)})^2} = \sqrt{(8-6)^2 + (7-6)^2} = \sqrt{5}$
- Can you calculate $D\left(\boldsymbol{x}^{(4)}, \boldsymbol{x}^{(2)}\right)$ and $D\left(\boldsymbol{x}^{(4)}, \boldsymbol{x}^{(3)}\right)$, and see which point $\boldsymbol{x}^{(4)}$ is closest to?
- What happens if k = 2? What if k = 3?

kNN algorithm

Input: neighbour size $k > 0$, training set $\{(\boldsymbol{x}^{(n)}, y^{(n)}): n = 1,2 \ldots N\}$, a new unlabelled data $\boldsymbol{x}^{(j)}$

for n = 1, 2… N // each example in the training set

Calculate $D(\boldsymbol{x}^{(j)}, \boldsymbol{x}^{(n)})$ // distance between $x^{(j)}$ *and* $x^{(n)}$

Select k training examples closest to $\boldsymbol{x}^{(j)}$

Return $y^{(j)}$ = the plurality vote of labels from the k examples.

(classification) or

$y^{(j)}$ = average/median of the y values of the k examples.

(regression)

Another visual example

kNN for a 3-class problem with k=5

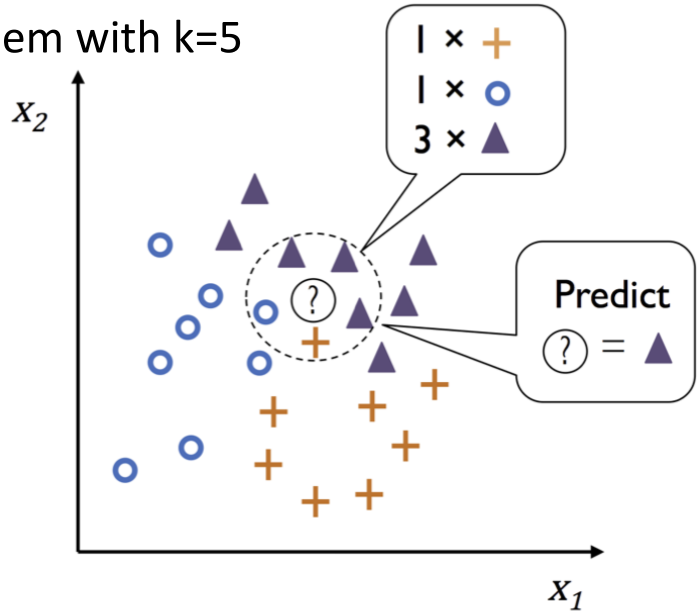


Image from “Python Machine Learning”, by Sybastian Raschka

Same example again

Consider a regression problem (lemon' weight) with 2 dimensions (height and width) with following training examples:

- $\boldsymbol{x}^{(1)}$ =(6,6), $y^{(1)}$ =10
- $\boldsymbol{x}^{(2)}$ =(8,10), $y^{(2)}$ =20
- $\boldsymbol{x}^{(3)}$ =(7,6), $y^{(3)}$ =15

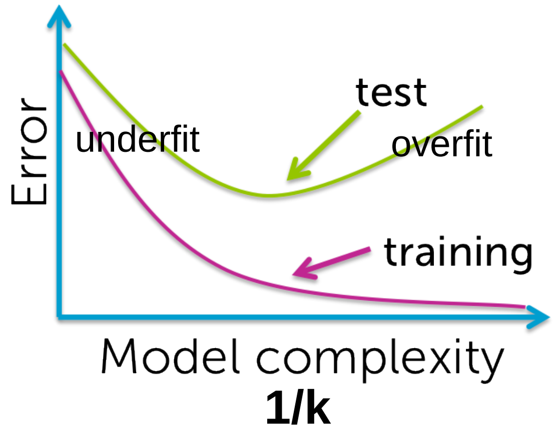
New example

- $\boldsymbol{x}^{(4)}$ =(8,7), $y^{(4)}$ =?
- $D\left(\boldsymbol{x}^{(4)},\boldsymbol{x}^{(1)}\right) = \sqrt{5}$, $D\left(\boldsymbol{x}^{(4)},\boldsymbol{x}^{(2)}\right) = 3$, $D\left(\boldsymbol{x}^{(4)},\boldsymbol{x}^{(3)}\right) = \sqrt{2}$
- If k = 3, and $\boldsymbol{x}^{(1)}$, $\boldsymbol{x}^{(2)}$, $\boldsymbol{x}^{(3)}$ are the closest 3 points to $\boldsymbol{x}^{(4)}$, what is the label of $\boldsymbol{x}^{(4)}$?

- $y^{(4)}$ = (10+20+15)/3 = 15

How to choose k?

- Recall: Overfitting and Underfitting
- k changes model complexity: smaller k -> higher complexity



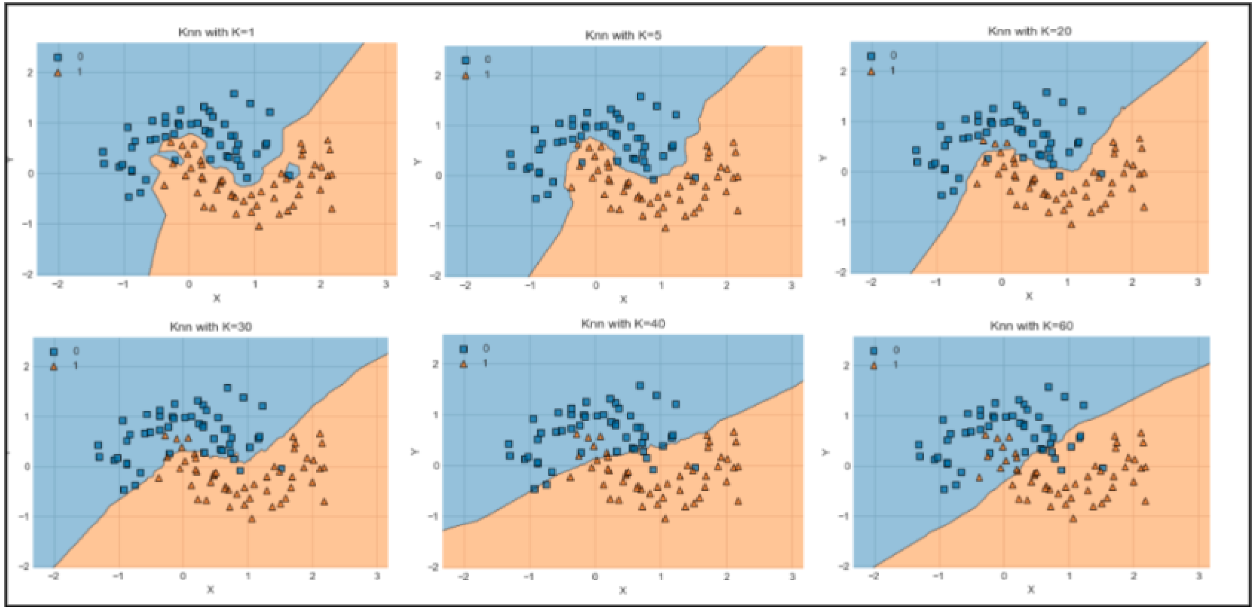


Image from https://neptune.ai/blog/knn-algorithm-explanation-opportunities-limitations

How to choose k?

- Small k -> small neighborhood -> high complexity -> may overfit
- Large k -> large neighborhood -> low complexity -> may underfit
- Practicians often choose k between 3 – 15, or k < $\sqrt{N}$ (N is the number of training examples).
- Refer to “model selection/evaluation” to be learnt next week.

The issue in numeric attribute ranges

- Attributes $\boldsymbol{x} = (x_1, x_2, \ldots, x_d)$ may have different ranges.
- The attribute with a larger range is treated as more important by the kNN algorithm (*some learning bias is embedded*!)
- It can affect the performance if you don't want to treat attributes differently.
- For example, if x_1 is in [0, 2] (e.g. height), and x_2 is in [0, 100] (e.g. age), x_2 will affect the distance more.
- Solutions? Normalisation

Normalisation and Standardization

- Method 1 Normalisation: Linearly scale the range of each attribute to be, e.g. in [0,1].

$$x_{j\_new}^{(n)} = \frac{x_j^{(n)} - \min x_j}{\max x_j - \min x_j}$$

- Method 2 Standardization: Linearly scale each dimension to have 0 mean and variance 1 (by computing mean μ and variance σ^2).

$$x_{j\_new}^{(n)} = \frac{x_j^{(n)} - \mu_j}{\sigma_j}, \text{ where } \mu_j = \frac{1}{N}\sum_{n=1}^{N} x_j^{(n)}, \sigma_j = \sqrt{\frac{1}{N}\sum_{n=1}^{N}(x_j^{(n)} - \mu_j)^2}$$

Example

- Consider a dataset with 2 dimensions (i.e. Attributes), where x_1 represents the age of a patient and x_2 represents the body weight. The output $y \in \{normal, abnormal\}$.

Patient	x_1	x_2	y
$\boldsymbol{x}^{(1)}$	14	70	n
$\boldsymbol{x}^{(2)}$	12	90	a
$\boldsymbol{x}^{(3)}$	15	66	n

- Normalize each attribute of $\boldsymbol{x}^{(1)}$ to [0,1].
- $x_{1\_new}^{(1)} = \frac{x_1^{(1)} - 12}{15-12} = \frac{14-12}{15-12} = 0.667$
- $x_{2\_new}^{(1)} = \frac{x_2^{(1)} - 66}{90-66} = \frac{70-66}{90-66} = 0.167$
- $\boldsymbol{x}^{(1)}$: (14, 70) -> (0.667, 0.167)
- $\boldsymbol{x}^{(2)}$: (12, 90) -> (0, 1)
- $\boldsymbol{x}^{(3)}$: (15, 66) -> (1, 0)

$x^{(4)}$: (16, 64)?

kNN algorithm with normalisation

Input: neighbour size $k > 0$, training set $\{(\boldsymbol{x}^{(n)}, y^{(n)}): n = 1,2 \dots N\}$, a new unlabelled data $\boldsymbol{x}^{(j)}$

Normalise/standardize $\boldsymbol{x}^{(j)} \rightarrow \boldsymbol{x}_{new}^{(j)}$

for n = 1, 2… N // each example in the training set

  Normalise/standardize $\boldsymbol{x}^{(n)} \rightarrow \boldsymbol{x}_{new}^{(n)}$

  Calculate $D(\boldsymbol{x}_{new}^{(j)}, \boldsymbol{x}_{new}^{(n)})$ // normalized/standardized distance

  Select k training examples closest to $x^{(j)}$

Return $y^{(j)}$ = the plurality vote of labels from the k examples.

(classification) or

$y^{(j)}$ = average/median of the y values of the k examples.

(regression)

Pros/cons

- kNN is a nonparametric, instance-based, lazy algorithm.
- Need to specify the distance function and pre-define k value.
- Easy to implement and interpret.
- It can approximate complex functions, so it has very good accuracy.
- It has to store all training data (large memory space), and calculate distance of each training example to the new example.

  There are smarter ways to store and use training data, e.g. KD-trees, remove redundant data.
- It can be sensitive to noise, especially when k is small.
- Its performance is degraded greatly as data dimension increases. (curse of dimensionality)

  As the volume grows larger, the “neighbors” become further apart and and not so close anymore. The prediction thus becomes less accurate.

Exercise

- x_1 and x_2 : numeric attributes, calculate distance as in previous examples.
- x_3 : ordinal attribute, discrete but has inherent ranking. {low, high} -> {0,1}
- x_4 : categorical attribute, hamming distance. {sunny, rainy}

	x_1	x_2	x_3	x_4	y
$\boldsymbol{x}^{(1)}$	0.4	73	high	sunny	1.6
$\boldsymbol{x}^{(2)}$	0.9	42	low	rainy	2.1
$\boldsymbol{x}^{(3)}$	0.1	61	high	rainy	1.9
$\boldsymbol{x}^{(4)}$	0.8	49	low	sunny	?

$$D\left(x^{(1)}, x^{(2)}\right) = \begin{cases} 0, & if\ x^{(1)} = x^{(2)} \\ 1, & otherwise \end{cases}$$

Requirement: k = 2, Manhattan distance

Exercise

- $\boldsymbol{x}^{(1)}$ =*{0.5, 1, 1, sunny}*
- $\boldsymbol{x}^{(2)}$ =*{1, 0, 0, rainy}*
- $\boldsymbol{x}^{(3)}$ =*{0, 0.613, 1, rainy}*
- $\boldsymbol{x}^{(4)}$ =*{0.875, 0.226, 0, sunny}*
- $D\left(\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(4)}\right)$ = 0.5 + 1 + 1 + 0 = 2.149
- $D\left(\boldsymbol{x}^{(2)}, \boldsymbol{x}^{(4)}\right)$ =1.351
- $D\left(\boldsymbol{x}^{(3)}, \boldsymbol{x}^{(4)}\right)$ = 3.262
- Traditional average: $y^{(4)}$=($y^{(1)}$+ $y^{(2)}$)/2 =(1.6+2.1)/2=1.85

Additional challenge: weighted kNN

Give more weights to the closer points

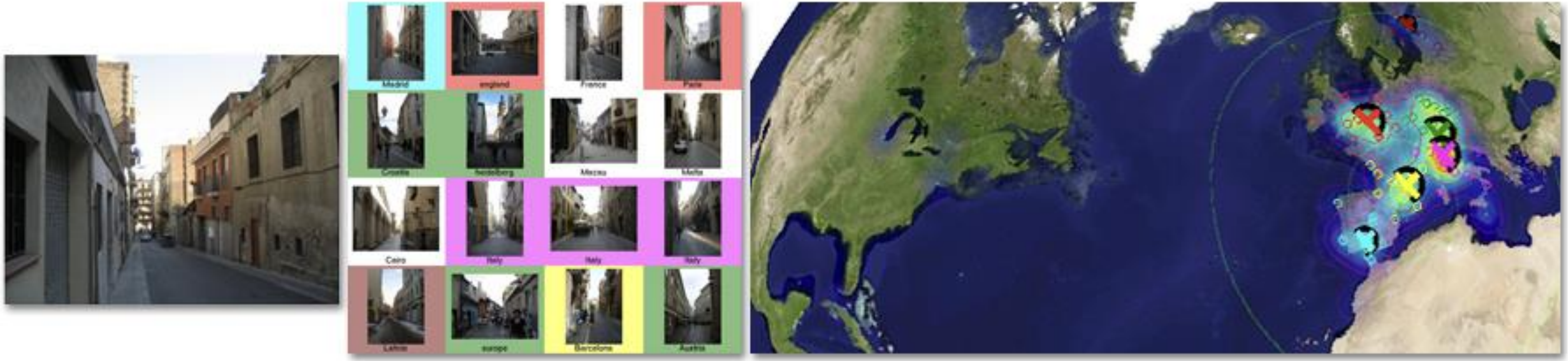
$$w_i = \frac{1}{D(\boldsymbol{x}^{(\boldsymbol{new})}, \boldsymbol{x}^{(\boldsymbol{i})})}$$

$$y^{(4)} = \frac{w_{\mathbf{1}} y^{(1)} + w_2 y^{(2)}}{w_{\mathbf{1}} + w_{\mathbf{2}}}$$

$y^{(4)}$=1.907, value towards $\boldsymbol{x}^{(2)}$

Fun project using kNN: where on earth is this photo from?

- Problem: where was this picture taken (country or GPS)?
- http://graphics.cs.cmu.edu/projects/im2gps/



- Get images from Flickr with gps info.
- Represent each image with meaningful features
- Apply kNN.

Q/A

Teams Channel: www.birmingham.ac.uk/
Office Hour: [faculty or individual email]@bham.ac.uk