AI1/AI&ML - k-Nearest Neighbours

Dr Leonardo Stella

Aims of the Session

This session aims to help you:

- Describe the steps of the k-Nearest Neighbours algorithm

- Apply k-Nearest Neighbours to problems involving numeric, ordinal and categorical input attributes

Overview

- **Notation**
- k-Nearest Neighbour
- k-NN algorithm and pros/cons

Notation

- Probabilistic models
  - Variables are denoted by uppercase letters, e.g., X or Y
  - Values that a variable can take are denoted by lowercase letters, e.g., x_1
  - Vectors are denoted by letters in bold, e.g., $\boldsymbol{X}$ or $\boldsymbol{x}$

Notation

- Probabilistic models
  - Variables are denoted by uppercase letters, e.g., X or Y
  - Values that a variable can take are denoted by lowercase letters, e.g., x_1
  - Vectors are denoted by letters in bold, e.g., $\boldsymbol{X}$ or $\boldsymbol{x}$

- In this lecture, we will be using the following notation:
  - Variables are denoted by lowercase letters, e.g., x or y
  - Values are typically stated, and letters are generally not used to represent values
  - Vectors are still denoted by letters in bold, e.g., $\boldsymbol{x}$

Nonparametric Models

- A nonparametric model is a model that cannot be characterised by a bounded set of parameters
  - For instance, suppose that each prediction we make will consider all training examples, including the one from the previous prediction(s)
  - The set of examples grows over time, thus nonparametric

Nonparametric Models

- A nonparametric model is a model that cannot be characterised by a bounded set of parameters
  - For instance, suppose that each prediction we make will consider all training examples, including the one from the previous prediction(s)
  - The set of examples grows over time, thus nonparametric
- This approach is also called **instance-** or **memory-based learning**
  - The simplest method for instance-based learning is table lookup
  - For table lookup, we put all training examples in a table, and when looking for a value, we return the corresponding value
  - Problem: if the value does not exist, then a default value is returned

Overview

- Notation
- **k-Nearest Neighbour**
- k-NN algorithm and pros/cons

k-Nearest Neighbours (k-NN or KNN)

- Consider the following two-dimensional problem (x_1, x_2)
  - Two classes: green or red $(y \in \{g, r\})$
  - New example (blue) to classify (majority vote)

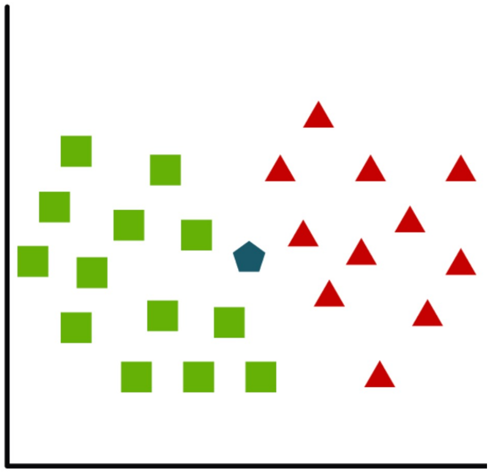

Image: taken from Ashraf *et al*., “A Review of Intrusion Detection Systems Using Machine and Deep Learning in Internet of Things: Challenges, Solutions and Future Directions”, *Electronics*, vol. 9, no. 7,2020.

k-Nearest Neighbours (k-NN or KNN)

- Consider the following two-dimensional problem (x_1, x_2)
  - Two classes: green or red $(y \in \{g, r\})$
  - New example (blue) to classify (majority vote)
  - Look at the k nearest neighbours
  - Let k = 3, to avoid issues
  - We want to predict the class of the new example

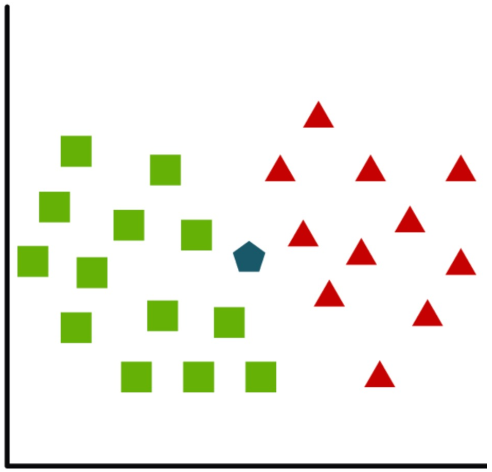

Image: taken from Ashraf *et al*., "A Review of Intrusion Detection Systems Using Machine and Deep Learning in Internet of Things: Challenges, Solutions and Future Directions", *Electronics*, vol. 9, no. 7,2020.

k-Nearest Neighbours (k-NN or KNN)

- Consider the following two-dimensional problem (x_1, x_2)
  - Two classes: green or red $(y \in \{g, r\})$
  - New example (blue) to classify (majority vote)
  - Look at the k nearest neighbours
  - Let k = 3, to avoid issues
  - We want to predict the class of the new example
  - In this case, we predict green
  - Intuitively, a distance metric on the **input space**

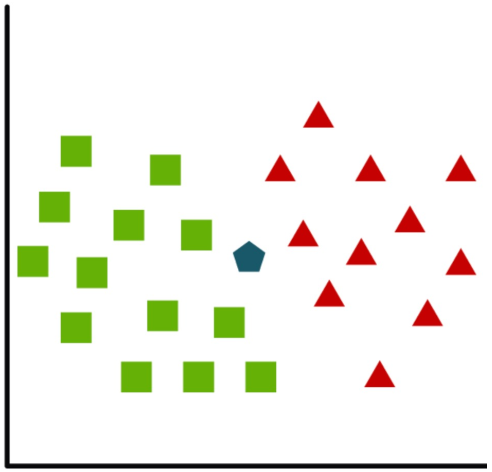

Image: taken from Ashraf *et al*., “A Review of Intrusion Detection Systems Using Machine and Deep Learning in Internet of Things: Challenges, Solutions and Future Directions”, *Electronics*, vol. 9, no. 7,2020.

Distance Metrics

- Consider a problem with n dimensions, $\boldsymbol{x}^{[q]}$ being the new example
- The Minkowski distance (or L^p norm) is defined as

$$L^p\left(\boldsymbol{x}^{[q]}, \boldsymbol{x}^{[i]}\right) = \sqrt[p]{\sum_{j=1}^{n} \left| x_j^{[q]} - x_j^{[i]} \right|^p}$$

- In general, the Euclidean distance is used, namely when $p = 2$

$$L^2\left(\boldsymbol{x}^{[q]}, \boldsymbol{x}^{[i]}\right) = \sqrt[2]{\sum_{j=1}^{n} (x_j^{[q]} - x_j^{[i]})^2}$$

Example

- Let us consider a problem with 2 dimensions (green and red), $\boldsymbol{x}^{[4]} = [0.1, 0.6]$ being the new example and the following training examples are given: $\boldsymbol{x}^{[1]} = [0.4, 0.2] \in \{green\}$, $\boldsymbol{x}^{[2]} = [0.4, 0.1] \in \{green\}$, $\boldsymbol{x}^{[3]} = [0.2, 0.6] \in \{red\}$

- Let us use the Euclidean distance to predict the new example

$$L^2\left(\boldsymbol{x}^{[4]}, \boldsymbol{x}^{[1]}\right) = \sqrt[2]{\sum_{j=1}^{n} (x_j^{[4]} - x_j^{[1]})^2} =$$

Example

- Let us consider a problem with 2 dimensions (green and red), $\boldsymbol{x}^{[4]} = [0.1, 0.6]$ being the new example and the following training examples are given: $\boldsymbol{x}^{[1]} = [0.4, 0.2] \in \{green\}$, $\boldsymbol{x}^{[2]} = [0.4, 0.1] \in \{green\}$, $\boldsymbol{x}^{[3]} = [0.2, 0.6] \in \{red\}$

- Let us use the Euclidean distance to predict the new example

$$L^2\left(\boldsymbol{x}^{[4]}, \boldsymbol{x}^{[1]}\right) = \sqrt[2]{\sum_{j=1}^{n} (x_j^{[4]} - x_j^{[1]})^2} = \sqrt{(x_1^{[4]} - x_1^{[1]})^2 + (x_2^{[4]} - x_2^{[1]})^2}$$

Example

- Let us consider a problem with 2 dimensions (green and red), $\boldsymbol{x}^{[4]} = [0.1, 0.6]$ being the new example and the following training examples are given: $\boldsymbol{x}^{[1]} = [0.4, 0.2] \in \{green\}$, $\boldsymbol{x}^{[2]} = [0.4, 0.1] \in \{green\}$, $\boldsymbol{x}^{[3]} = [0.2, 0.6] \in \{red\}$

- Let us use the Euclidean distance to predict the new example

$$L^2\left(\boldsymbol{x}^{[4]}, \boldsymbol{x}^{[1]}\right) = \sqrt[2]{\sum_{j=1}^{n} (x_j^{[4]} - x_j^{[1]})^2} = \sqrt{(x_1^{[4]} - x_1^{[1]})^2 + (x_2^{[4]} - x_2^{[1]})^2}$$

$$= \sqrt{(0.1 - 0.4)^2 + (0.6 - 0.2)^2} = \sqrt{0.09 + 0.16} = 0.5$$

Example

- Let us consider a problem with 2 dimensions (green and red), $\boldsymbol{x}^{[4]} = [0.1, 0.6]$ being the new example and the following training examples are given: $\boldsymbol{x}^{[1]} = [0.4, 0.2] \in \{green\}$, $\boldsymbol{x}^{[2]} = [0.4, 0.1] \in \{green\}$, $\boldsymbol{x}^{[3]} = [0.2, 0.6] \in \{red\}$

- We repeat the process for all the points:

$$L^2\left(\boldsymbol{x}^{[4]}, \boldsymbol{x}^{[1]}\right) = 0.5$$
$$L^2\left(\boldsymbol{x}^{[4]}, \boldsymbol{x}^{[2]}\right) = 0.583$$
$$L^2\left(\boldsymbol{x}^{[4]}, \boldsymbol{x}^{[3]}\right) = 0.1$$

Example

- Let us consider a problem with 2 dimensions (green and red), $\boldsymbol{x}^{[4]} = [0.1, 0.6]$ being the new example and the following training examples are given: $\boldsymbol{x}^{[1]} = [0.4, 0.2] \in \{green\}$, $\boldsymbol{x}^{[2]} = [0.4, 0.1] \in \{green\}$, $\boldsymbol{x}^{[3]} = [0.2, 0.6] \in \{red\}$

- We repeat the process for all the points:

$$L^2\left(\boldsymbol{x}^{[4]}, \boldsymbol{x}^{[1]}\right) = 0.5$$
$$L^2\left(\boldsymbol{x}^{[4]}, \boldsymbol{x}^{[2]}\right) = 0.583$$
$$L^2\left(\boldsymbol{x}^{[4]}, \boldsymbol{x}^{[3]}\right) = 0.1$$

k-NN for Regression Problems

- Consider the following two-dimensional problem (x_1, x_2)
  - Let us consider examples that take a value between 1 and 5 ($y \in [1, 5]$)
  - Assume each example has a value, e.g., 3.2 or 4.1
  - Look at the k nearest neighbours
  - We want to predict the value of the new example
  - In this case, we predict the average or median of the values of the k nearest neighbours

Overview

- Notation
- k-Nearest Neighbour
- **k-NN algorithm and pros/cons**

k-NN Algorithm

- Input: training examples $x^{[i]} \in \boldsymbol{x}$ and their corresponding class $y^{[i]}$, a new query example $x^{[q]}$, number of neighbours k
- Output: prediction of the new query example $x^{[q]}$
- For each training example $x^{[i]} \in \boldsymbol{x}$
  - Calculate the distance between the training example $x^{[i]}$ and the new query example $x^{[q]}$
  - Keep the best k distances (the shortest distance) in a data structure T
- Return the majority vote (or average/median) of the class $y^{[i]}$ for the first k entries of T

Problem with Numeric Independent Variables

- Different numeric attributes may have different scales
- For example, if x_1 is in [0,1] and x_2 is in [1, 10], x_2 will affect the distance more

Problem with Numeric Independent Variables

- Different numeric attributes may have different scales
- For example, if x_1 is in [0,1] and x_2 is in [1, 10], x_2 will affect the distance more
- To avoid this problem, we normalise the numeric input attributes of all data as in the following

$$normalise\left(x_j^{[i]}\right) = \frac{x_j^{[i]} - \min_j}{\max_j - \min_j}$$

- Another approach (see book) is to calculate mean μ_j and standard deviation σ_j for each dimension j as: $(x_j^{[i]} - \mu_j)/\sigma_j$

Example: Normalisation

- Consider the following $\boldsymbol{x}^{[1]} = [170, 50] \in \{yes\}$, where $x_1 = Age$ and $x_2 = Weight, y \in \{yes, no\}$

- Let us calculate the normalised values for $\boldsymbol{x}^{[1]}$:
- $normalise\left(x_1^{[1]}\right) = \frac{x_1^{[1]} - \min_1}{\max_1 - \min_1} = \frac{14 - 12}{15 - 12} = 0.667$
- $normalise\left(x_2^{[1]}\right) = \frac{x_2^{[1]} - \min_2}{\max_2 - \min_2} = \frac{70 - 66}{90 - 66} = 0.167$

Days	x_1	x_2	y
$\boldsymbol{x}^{[1]}$	14	70	yes
$\boldsymbol{x}^{[2]}$	12	90	no
$\boldsymbol{x}^{[3]}$	15	66	yes

k-NN Algorithm with Normalisation

- Input: training examples $x^{[i]} \in \boldsymbol{x}$ and their corresponding class $y^{[i]}$, a new query example $x^{[q]}$, number of neighbours k
- Output: prediction of the new query example $x^{[q]}$
- For each training example $x^{[i]} \in \boldsymbol{x}$
  - Calculate the normalised distance between the training example $x^{[i]}$ and the new query example $x^{[q]}$
  - Keep the best k distances (the shortest distance) in a data structure T
- Return the majority vote (or average/median) of the class $y^{[i]}$ for the first k entries of T

Different Input Attributes

- For numeric input attributes, e.g., age in [0, 100], we calculate the distance as shown in previous examples
- For ordinal input attributes, e.g., sunny in {yes, no}, we can convert the values to numeric values: yes = 1, no = 0
- For categorical input attributes, e.g., phone_brand in {samsung, apple, nokia}, we can use the following approach:
  - If the value of the query example is the same as the value for example i, then their difference is 0. Formally, if $\left(x_j^{[q]} = x_j^{[i]}\right)$, then $\left(x_j^{[q]} - x_j^{[i]}\right) = 0$
  - Otherwise, their difference is 1. Formally, if $\left(x_j^{[q]} \neq x_j^{[i]}\right)$, then $\left(x_j^{[q]} - x_j^{[i]}\right) = 1$

Summary

k-NN Learning Algorithm

- The algorithm does not have proper training
- We simply store all training data, which increase over time
- We normalise by calculating the minimum and maximum in the training data

k-NN Model

- All training data, the values of the numeric input attributes

k-NN prediction for an instance ($\boldsymbol{x}^{[i]}, y = ?$)

- Find the k nearest neighbours whose distance to $\boldsymbol{x}^{[i]}$ is the smallest
- For classification problems, majority vote. For regression problems, average/median

Pros and Cons of k-NN

Pros

- Training is simple and fast: just store training data
- Find the class of the new example based on most similar examples present in the training data

Cons

- It uses large space in memory: we need to store all data
- Running the algorithm can be slow if we have many training examples and many dimensions

Aims of the Session

You should now be able to:

- Describe the steps of the k-Nearest Neighbours algorithm

- Apply k-Nearest Neighbours to problems involving numeric, ordinal and categorical input attributes