CO3091 - Computational Intelligence and Software Engineering

Lecture 16

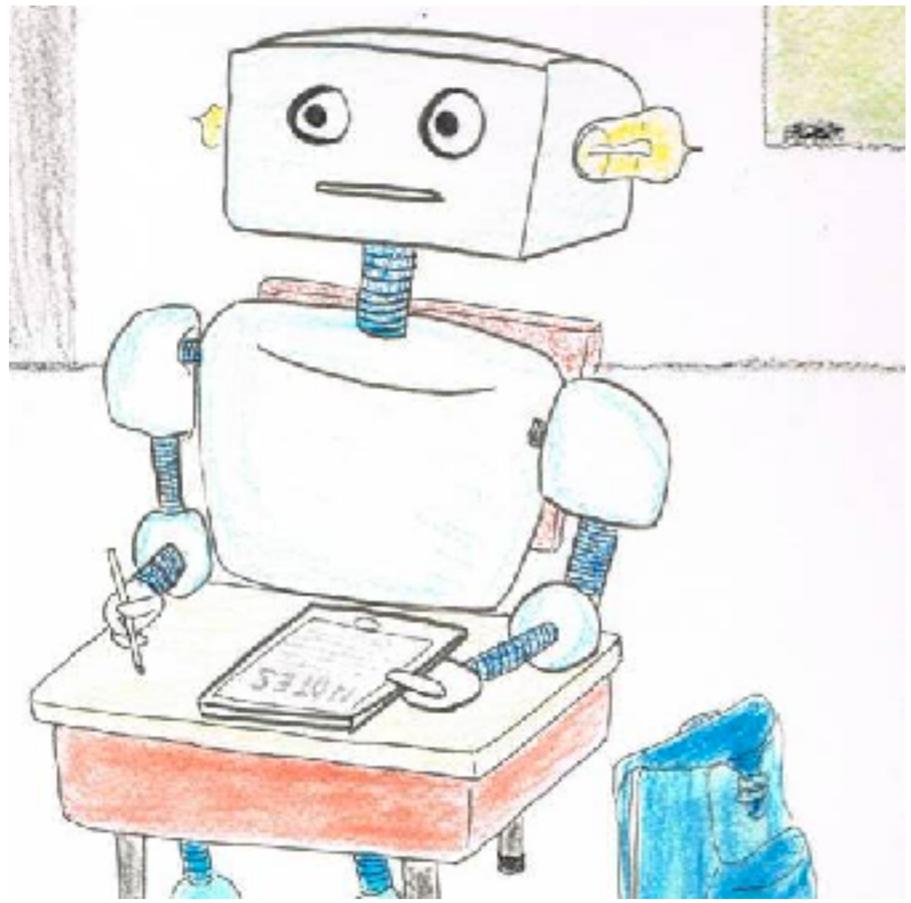


Image from: http://gureckislab.org/blog/wp-content/uploads/2012/09/Machine-Learning_drawing_square1.jpg

Introduction to Machine Learning and k-NN

Leandro L. Minku

Announcements

- Coursework 2 out!

Overview

- Introduction to machine learning
- Software effort estimation
- k-NN (k-Nearest Neighbours)

Machine Learning

Focus: to study and develop computational models capable of improving their performance with experience and acquiring knowledge on their own.

- How?
  - Through data examples (a.k.a. data points, instances).
  - Types of learning:
    - supervised learning,
    - unsupervised learning,
    - semi-supervised learning and
    - reinforcement learning.

Image from: http://bioap.wikispaces.com/file/view/social_learngin.jpg/101074725/social_learngin.jpg

Example of Problem That Can be Solved Using Supervised Learning

Software effort estimation:

- Estimation of the effort required to develop a software project.
    - Effort is measured in person-hours, person-months, etc.
- Based on features such as programming language, team expertise, estimated size, development type, required reliability, etc.
- Main factor influencing project cost.
- Overestimation vs underestimation.

Example of Underestimation

Nasa cancelled its incomplete Check-out Launch Control Software project after the initial $200M estimate was exceeded by another $200M.

Machine Learning for Software Effort Estimation

- Software effort estimation is difficult to perform by humans.
  - Affected by irrelevant features.
  - Lack of improvement in the predictions over time.
- Supervised learning can help.
  - Predictive models can be created based on data describing past projects and their required efforts.
  - These predictive models can be used as decision-support tools to predict the effort for new projects.

Supervised Learning

- Predictive tasks: based on existing [training] data, learn models able to make predictions for new data.

- A data example has the format (**x**,y), where
  - **x** are the input attributes (a.k.a. input features, independent variables).
    - We have n input attributes: $\mathbf{x} = (x_1, x_2, \ldots, x_n)$.

  - y is the output attribute (a.k.a. output feature, target values, label, class, dependent variable).
    - PS: some problems have more than one output attribute, but we will be considering problems with a single output attribute here.

Supervised Learning Problem

- E.g.: software effort estimation

	x_1 (programming language)	x_2 (team expertise)	x_3 (estimated size)	…	y (required effort)
$(\mathbf{x_1}, y_1)$	x_{11}	x_{12}	x_{13}	…	10 p- y_1 onth
$(\mathbf{x_2}, y_2)$	x_{21}	x_{22}	x_{23}	…	20 p- y_2 onth
$(\mathbf{x_3}, y_3)$	x_{31}	x_{32}	x_{33}	…	8 p- y_3 onth
…	…	…	…	…	…

Supervised Learning Problem

- E.g.: credit card approval

	x_1 (age)	x_2 (salary)	x_3 (gender)	…	y (good/bad payer)
$(\mathbf{x_1}, y_1)$	18	1000	female	…	Good
$(\mathbf{x_2}, y_2)$	30	900	male	…	Bad
$(\mathbf{x_3}, y_3)$	20	5000	female	…	Good
…	…	…	…	…	…

Types of Input Attributes

Numerical:

- E.g., age, salary.

Ordinal:

- E.g., expertise in {low, medium, high}.

Categorical:

- E.g., car in {fiat, volkswagen, toyota}.

Classification vs Regression

- Classification problem:
  - The output attributes are categories / classes.
  - E.g.: good or bad payer.
  - A predictive model for a classification problem is frequently referred to as a classifier.

- Regression problem:
  - The output attributes are numerical values.
  - E.g.: how much effort we would require to develop a software project.

Supervised Learning

[Pre-processed]
Training Data / Examples

x_1 (age)	x_2 (salary)	x_3 (gender)	…	y (good/bad payer)
18	1000	female	…	Good
30	900	male	…	Bad
20	5000	female	…	Good
…	…	…	…	…

→ Machine Learning Algorithm →

Predictive Model

New instance **x** for which we want to predict the output →

→ Prediction

Image from: http://www.theenergycollective.com/sites/theenergycollective.com/files/imagepicker/488516/2013%20predictions%20follow%20up.jpg
Image from: http://api.ning.com/files/jPrLLXuRuu5NQUBB0TWzUm-twYVH7ySIXAqNfxr1znmJjEDubI*jqwrO0EfEEgHXuDo41tbZK*Nk1rYqydv-PiWaKyjdpQt6/2013Predictions.jpg

Machine Learning Problem

[Pre-processed]
Training Data / Examples

x_1 (age)	x_2 (salary)	x_3 (gender)	…	y (good/bad payer)
18	1000	female	…	Good
30	900	male	…	Bad
20	5000	female	…	Good
…	…	…	…	…

→ Machine Learning Algorithm →

Predictive Model

New instance **x** for which we want to predict the output →

→ Prediction

Image from: http://www.theenergycollective.com/sites/theenergycollective.com/files/imagepicker/488516/2013%20predictions%20follow%20up.jpg
Image from: http://api.ning.com/files/jPrLLXuRuu5NQUBB0TWzUm-twYVH7ySIXAqNfxr1znmJjEDubI*jqwrO0EfEEgHXuDo41tbZK*Nk1rYqydv-PiWaKyjdpQt6/2013Predictions.jpg

Components of a Machine Learning Approach

[Pre-processed]
Training Data / Examples

x_1 (age)	x_2 (salary)	x_3 (gender)	…	y (good/bad payer)
18	1000	female	…	Good
30	900	male	…	Bad
20	5000	female	…	Good
…	…	…	…	…

→ Machine Learning Algorithm →

Predictive Model

New instance **x**
for which we want to predict the output →

→ Prediction

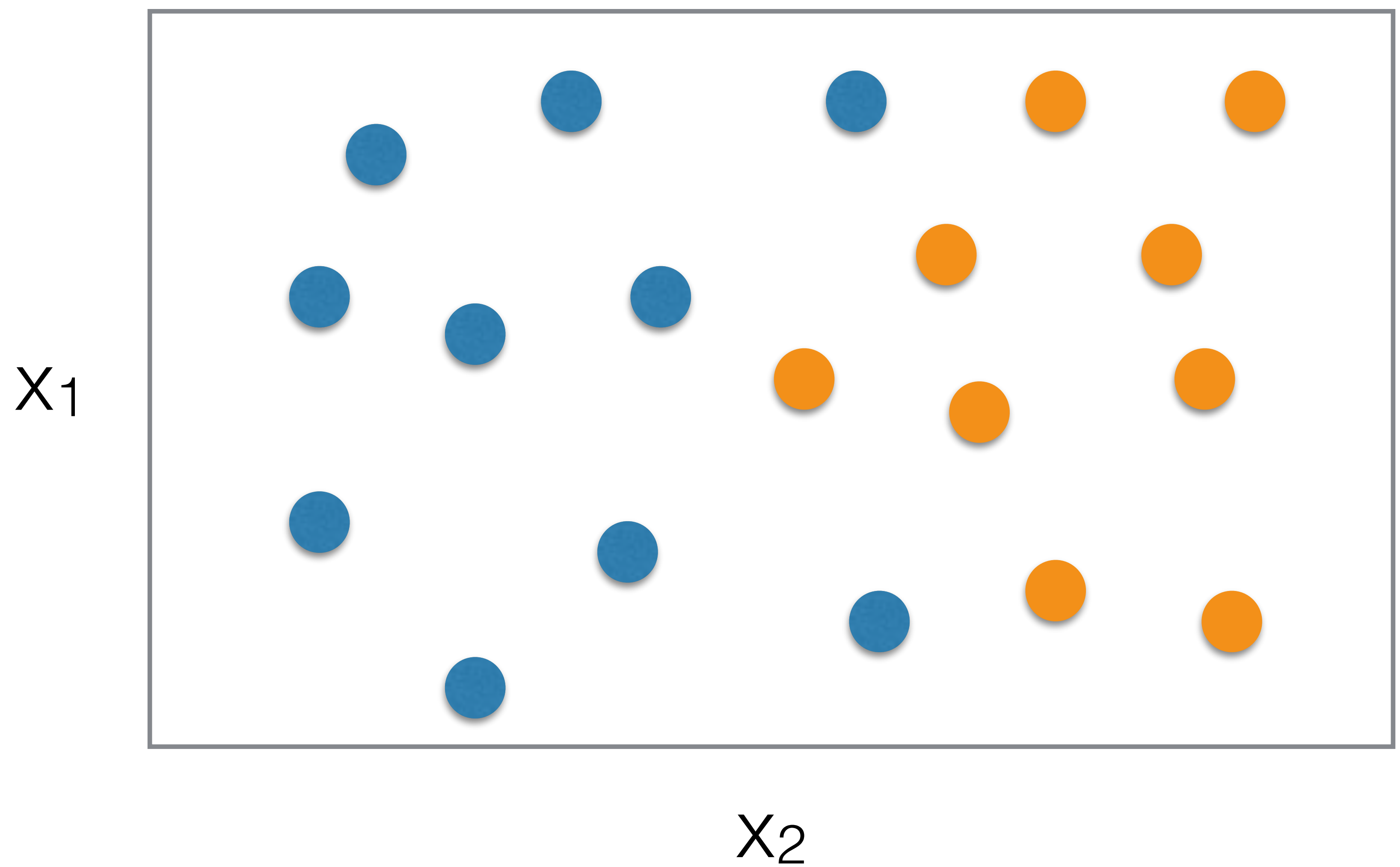
Image from: http://www.theenergycollective.com/sites/theenergycollective.com/files/imagepicker/488516/2013%20predictions%20follow%20up.jpg
Image from: http://api.ning.com/files/jPrLLXuRuu5NQUBB0TWzUm-twYVH7ySIXAqNfxr1znmJjEDubI*jqwrO0EfEEgHXuDo41tbZK*Nk1rYqydv-PiWaKyjdpQt6/2013Predictions.jpg

How do Machine Learning Approaches Look Like?

- There are many different types of machine learning approaches.

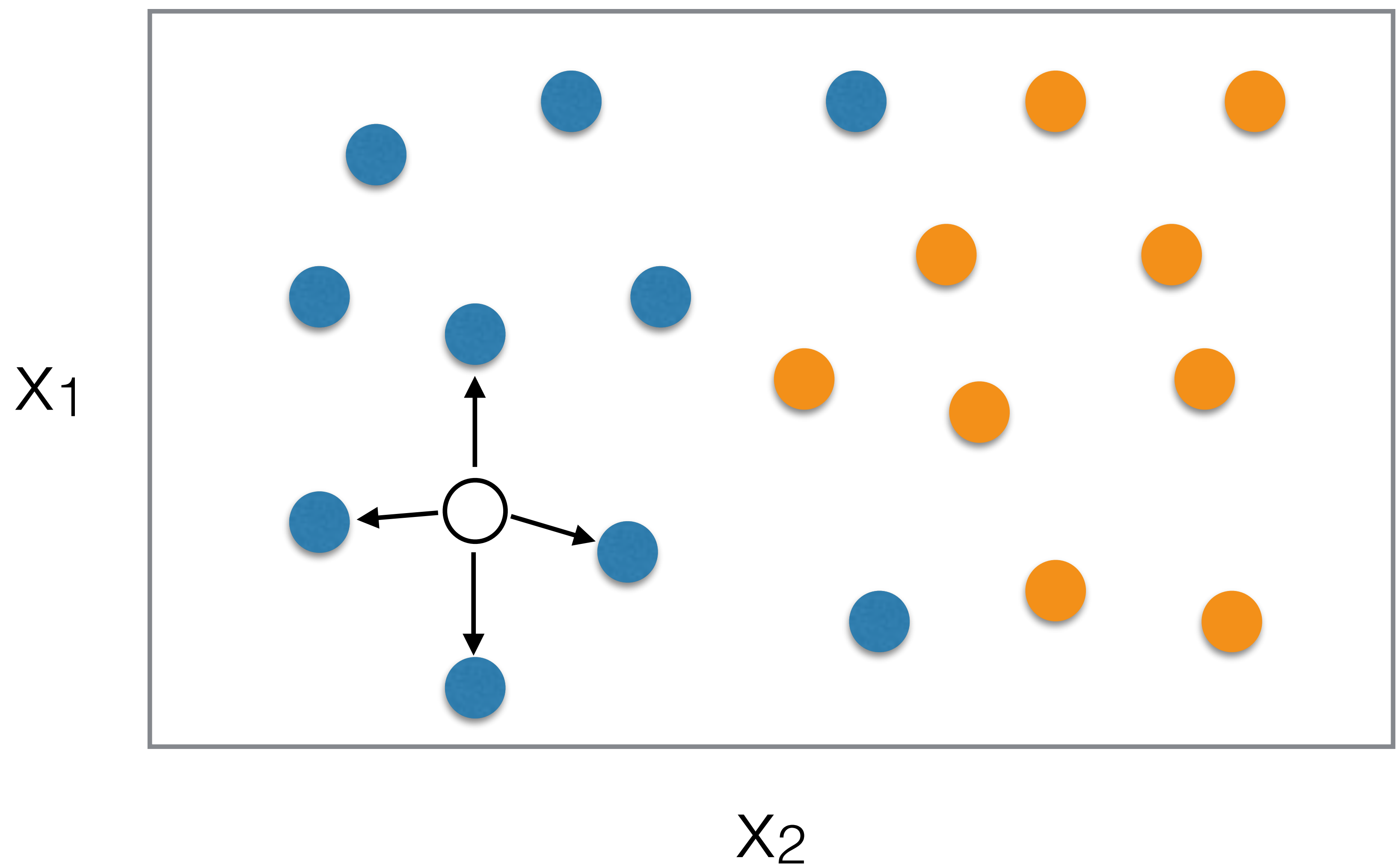
- k-NN (k-Nearest Neighbour) is an example of machine learning approach.

k-Nearest Neighbours: Basic Idea



Targets: y=blue or y=orange

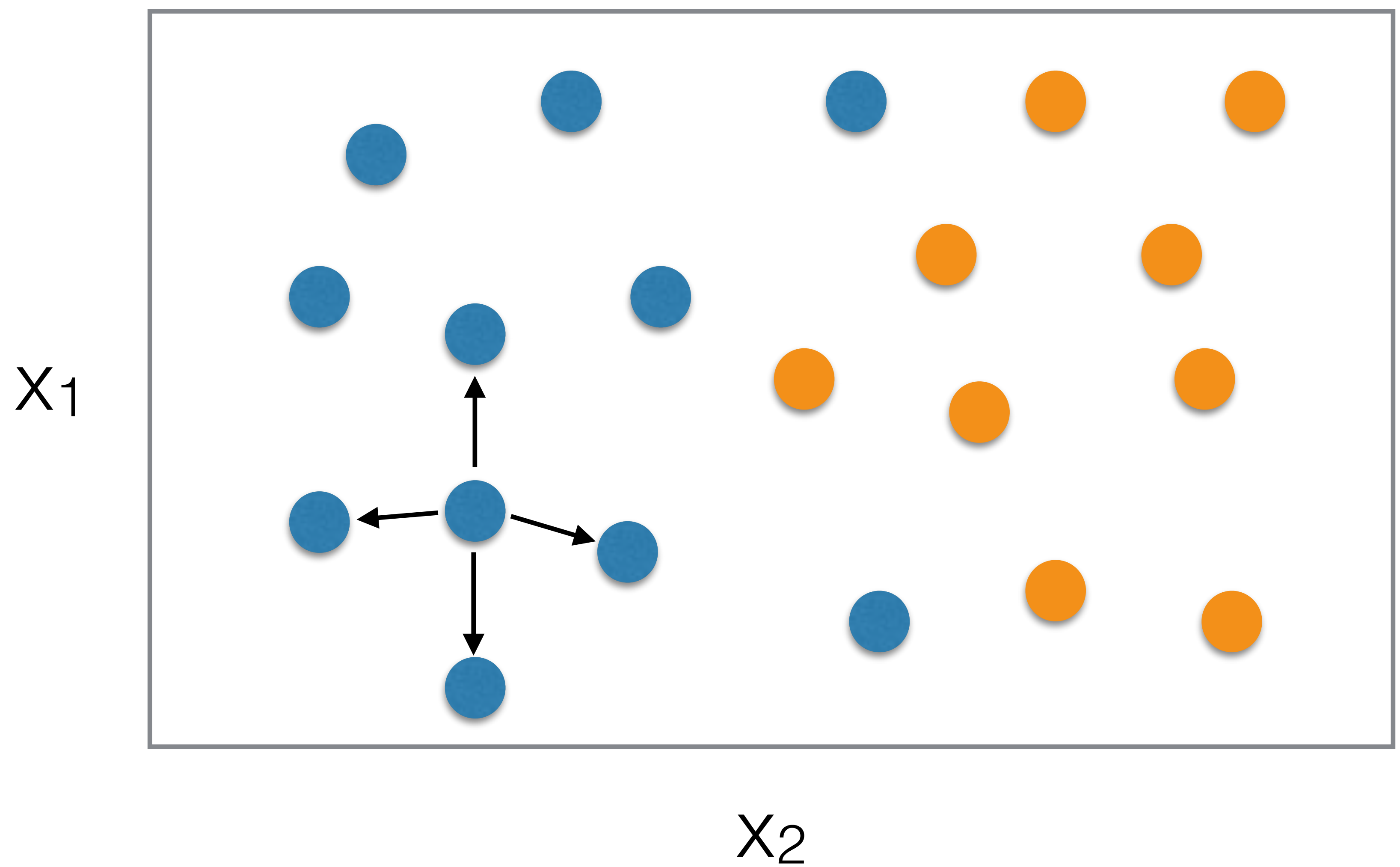
k-Nearest Neighbours: Basic Idea



K = 4

Usually, for classification problems: predict the majority among the output attributes of the nearest neighbours (majority vote).

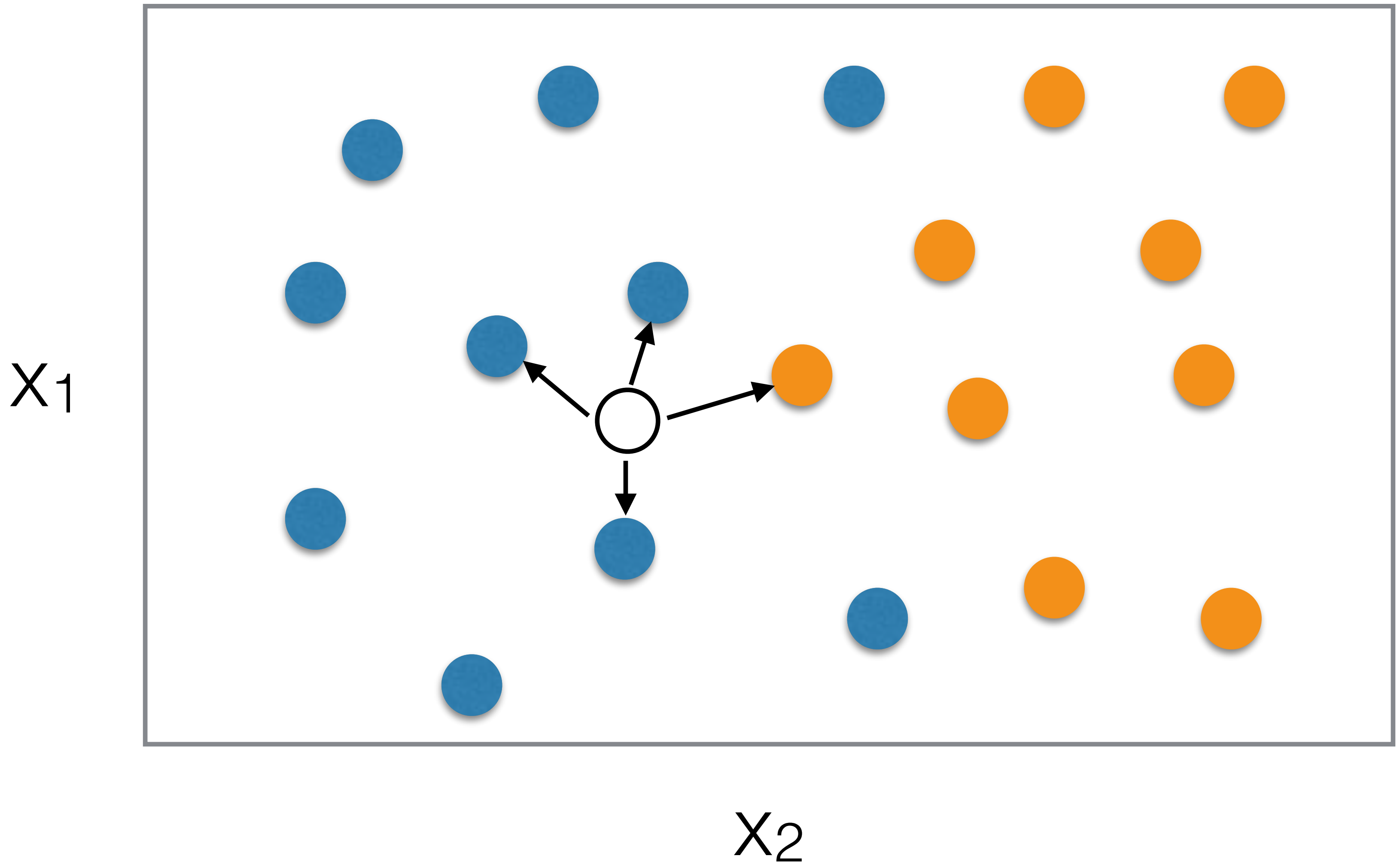
k-Nearest Neighbours: Basic Idea



K = 4

Usually, for classification problems: predict the majority among the output attributes of the nearest neighbours (majority vote).

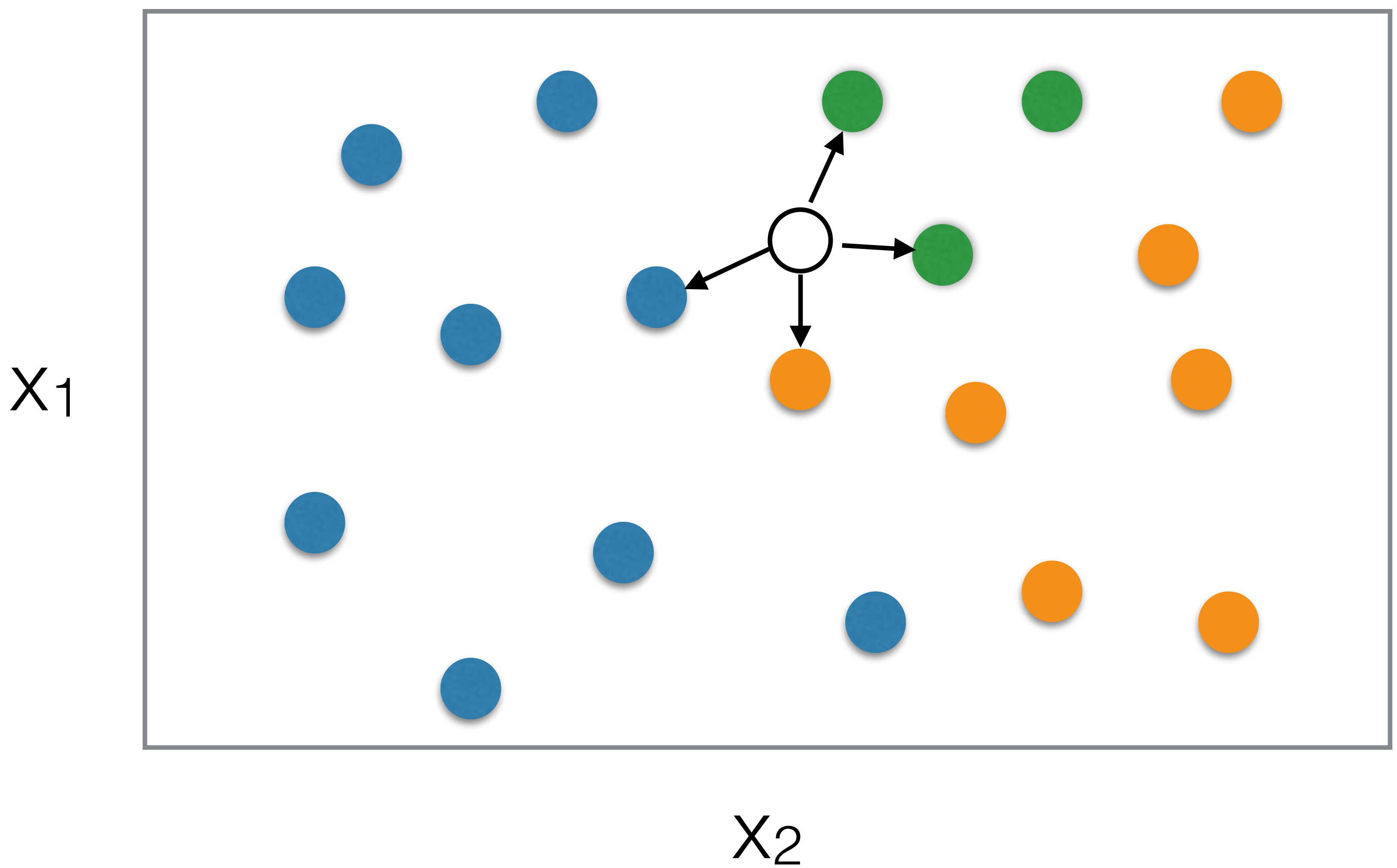
k-Nearest Neighbours: Basic Idea



K = 4

What is the predicted output for this instance?

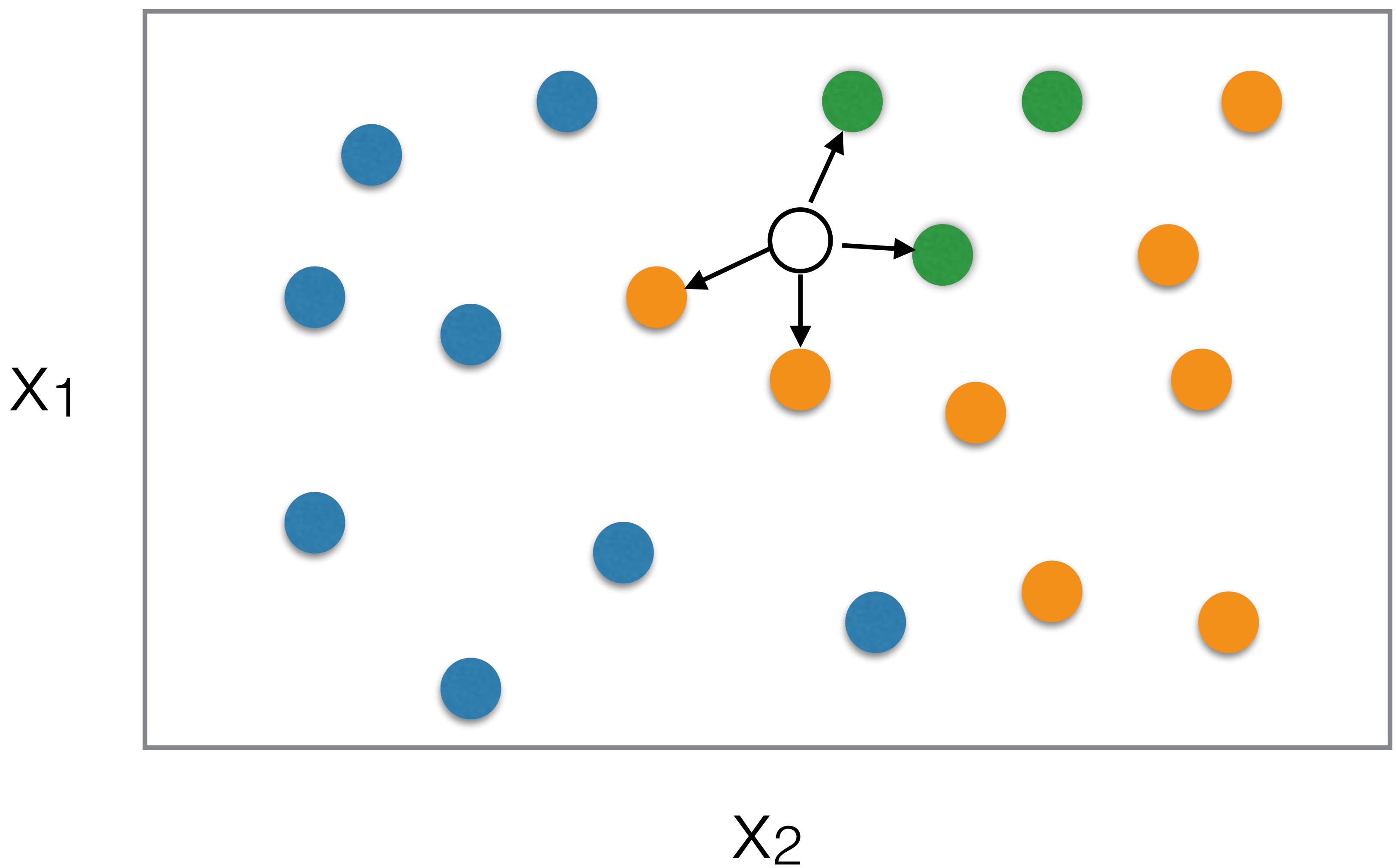
k-Nearest Neighbours: Basic Idea



K = 4

What is the predicted output for this instance?

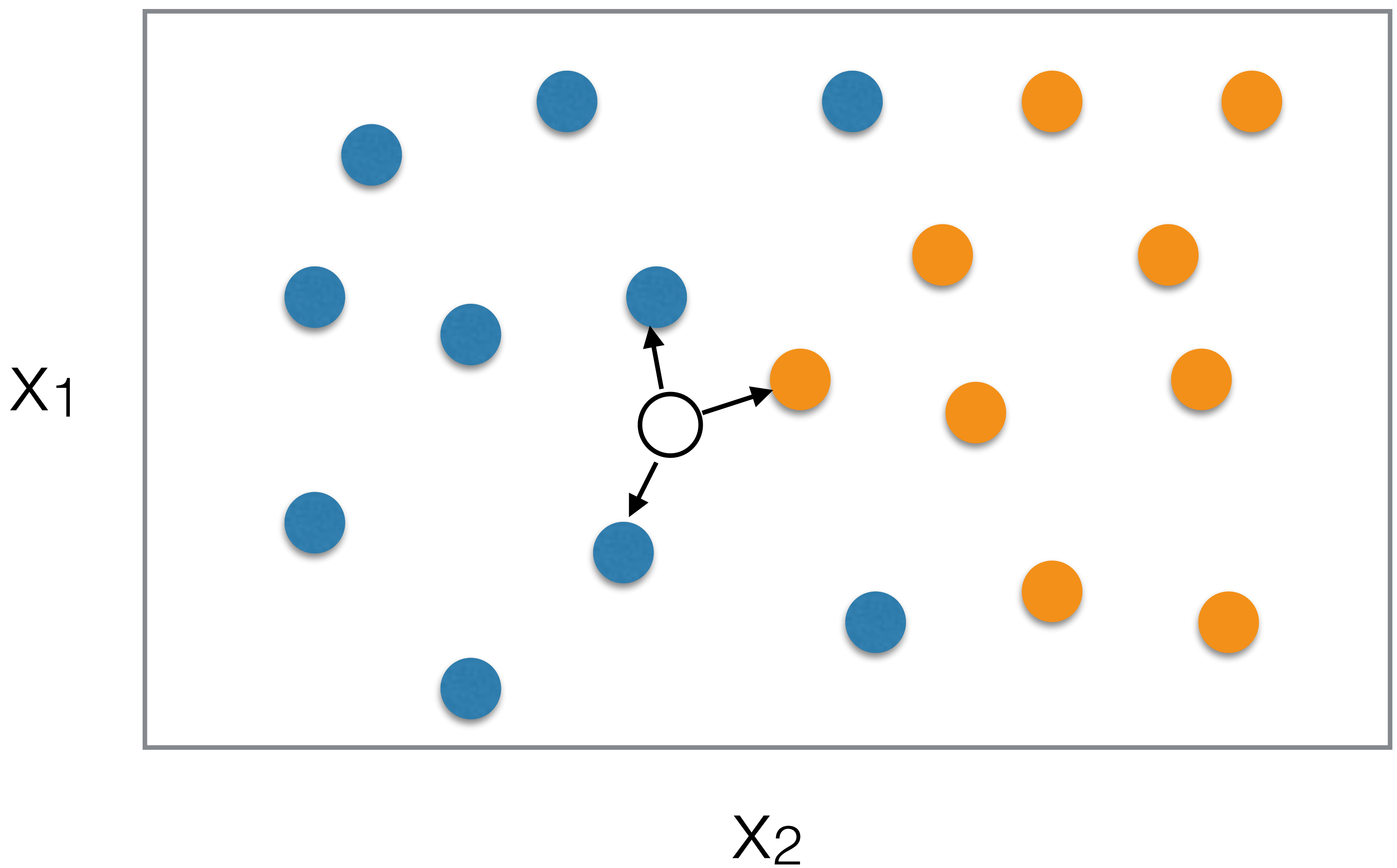
k-Nearest Neighbours: Basic Idea



K = 4

What is the predicted output for this instance?

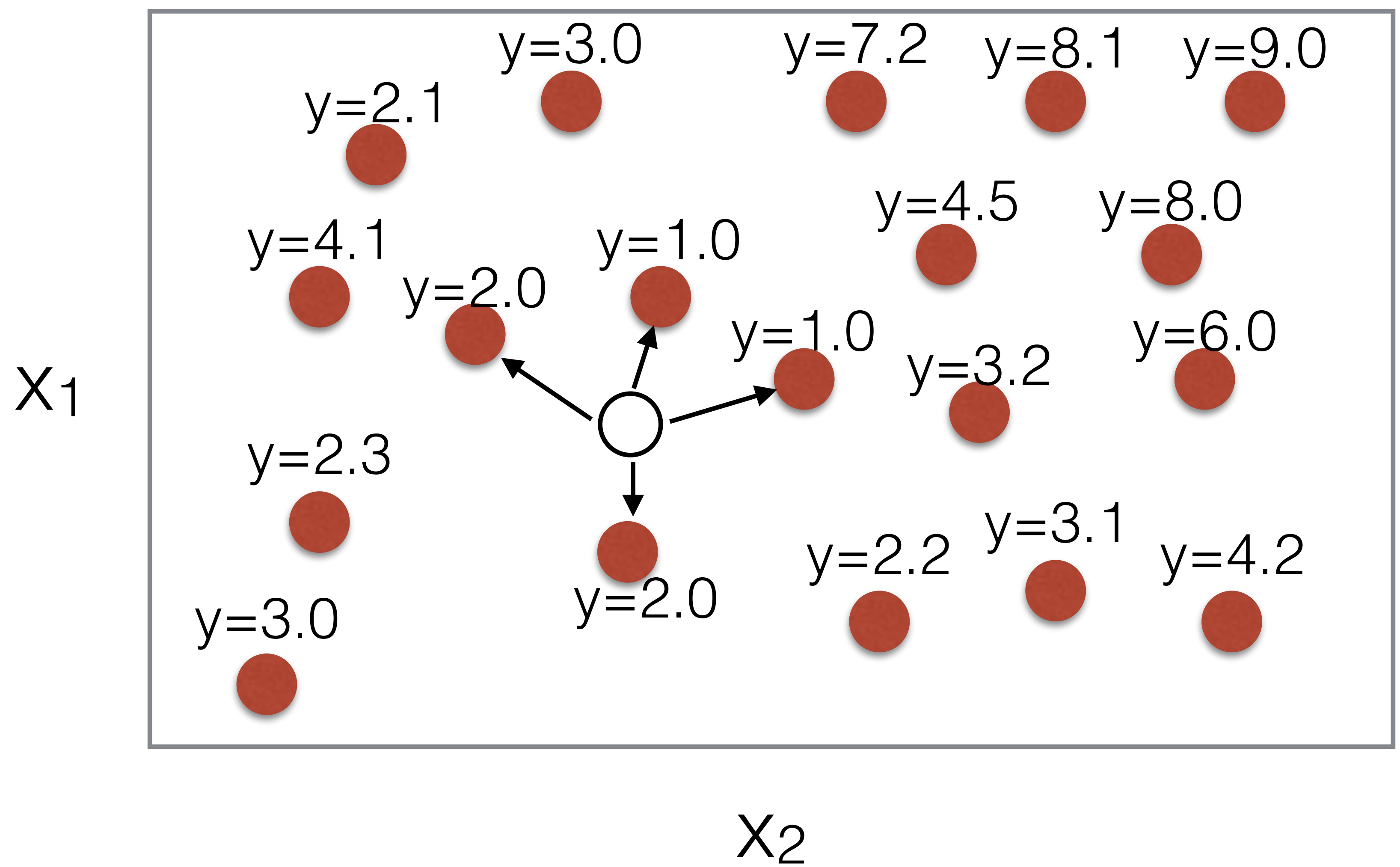
k-Nearest Neighbours: Basic Idea



K = 3

What is the predicted output for this instance?

k-Nearest Neighbours: Basic Idea



K = 4

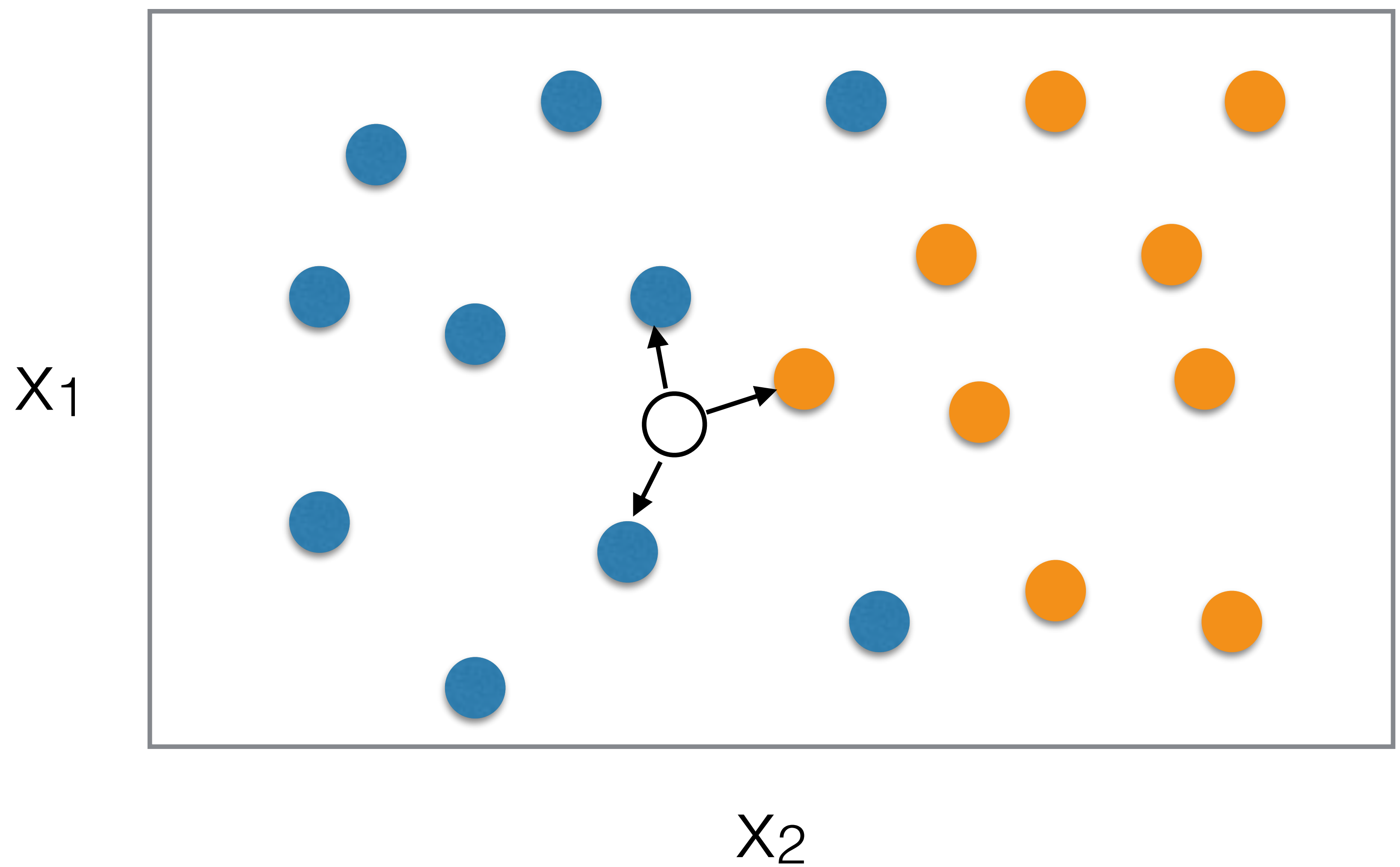
Usually, for regression problems: predict the average among the outputs of the nearest neighbours.

k-Nearest Neighbours: Basic Idea

Assumption: data points that are close to each other in the input space are also close to each other in the output space.

Image from: http://2.bp.blogspot.com/-LW4n9YDzoOA/UuJ2Idz-FyI/AAAAAAAAF1I/GUao58T9KjI/s1600/app-rejected.png

k-Nearest Neighbours: Basic Idea



K = 3

Given an instance to be predicted, we need to find its k nearest neighbours.

Finding the K Nearest Neighbours

- Usually, this is based on the Euclidean Distance **in the input space.**

- For n dimensions in the input space:

$$\text{distance}(\mathbf{x},\mathbf{x'}) = \sqrt{(x_1 - x'_1)^2 + (x_2 - x'_2)^2 + \ldots + (x_n - x'_n)^2}$$

where n is the number of input attributes

$$\text{distance}(\mathbf{x},\mathbf{x'}) = \sqrt{\sum_{i=1}^{n} (x_i - x'_i)^2}$$

Normalisation of Input Attributes

- Problem: different input attributes may have different scales.
  - Scale of input attributes will influence the Euclidean Distance.
  - If x_1 is in [0,10] and x_2 is in [100,10000], x_2 will influence the distance more.

- Popular solution:
  - Normalise input attributes of all data so that they will be between 0 and 1. E.g.:

$$\text{normalised}(x_i) = \frac{x_i - \text{min}_i}{\text{max}_i - \text{min}_i}$$

What is the normalised value of $x_1 = 5$?

Normalisation of Input Attributes

- How to know the minimum and maximum values?
  - If the real minimum and maximum are unknown, for each input attribute, use the minimum and maximum values present in the training set.
  - If later on you find new minimum or maximum values, you need to de-normalise and re-normalise the data.

+ * $$\text{normalised}(x_i) = \frac{x_i - \min_i}{(\max_i - \min_i)}$$

Ordinal or Categorical Input Attributes

- Input attributes can be numerical, ordinal or categorical.
  - Numerical: e.g., age, salary.
  - Ordinal: e.g., expertise in {low, medium, high}.
  - Categorical: e.g., car in {fiat, volkswagen, toyota}.

- Euclidean distance is defined for numerical data!

$$\text{distance}(\mathbf{x},\mathbf{x}') = \sqrt{\sum_{i=1}^{n} (x_i - x'_i)^2}$$

- For ordinal input attributes, we can convert them to numerical.
  - E.g.: low = 0, medium = 0.5, high = 1.

- For categorical input attributes, we could use the following idea:

$$\text{if } x_i = x'_i, \quad x_i - x'_i = 0$$
$$\text{if } x_i \neq x'_i, \quad x_i - x'_i = 1$$

Components of a Machine Learning Approach

[Pre-processed]
Training Data / Examples

x_1 (age)	x_2 (salary)	x_3 (gender)	…	y (good/bad payer)
18	1000	female	…	Good
30	900	male	…	Bad
20	5000	female	…	Good
…	…	…	…	…

→ Machine Learning Algorithm →

Predictive Model

New instance **x** for which we want to predict the output →

→ Prediction

Image from: http://www.theenergycollective.com/sites/theenergycollective.com/files/imagepicker/488516/2013%20predictions%20follow%20up.jpg
Image from: http://api.ning.com/files/jPrLLXuRuu5NQUBB0TWzUm-twYVH7ySIXAqNfxr1znmJjEDubI*jqwrO0EfEEgHXuDo41tbZK*Nk1rYqydv-PiWaKyjdpQt6/2013Predictions.jpg

k-NN Approach

- k-NN Learning Algorithm:
  - No real training; simply normalise and store all training data received so far, together with their maximum and minimum numerical input attribute values
- k-NN "Model":
  - All normalised training data received so far, together with their maximum and minimum numerical input attribute values.
- k-NN prediction for an instance (**x**,?):
  - Find the K nearest neighbours, i.e., the K training examples that are the closest to **x.**
  - For classification problems: majority vote.
  - For regression problems: average.

Advantages and Disadvantages

- Advantages:
  - Training is simple and quick: just store the training data (possibly after some pre-processing).

- Disadvantage:
  - Memory requirements are high: stores all data, which can be troublesome when training set is large.
  - Making predictions is slow: we have to search for the nearest neighbours among all the training data, which can be troublesome when training set is large.

[Original] k-NN is not adequate when we have very large training sets.

Advantages and Disadvantages

- Advantages:
  - Training is simple and quick: just store the training data (possibly after some pre-processing).
- Disadvantage:
  - Memory requirements are high: stores all data, which can be troublesome when training set is large.
  - Making predictions is slow: we have to search for the nearest neighbours among all the training data, which can be troublesome when training set is large.

k-NN can be good for applications where there is little data, e.g.: software effort estimation.

Advantages and Disadvantages

- Advantages:
  - Training is simple and quick: just store the training data (possibly after some pre-processing).
- Disadvantage:
  - Memory requirements are high: stores all data, which can be troublesome when training set is large.
  - Making predictions is slow: we have to search for the nearest neighbours among all the training data, which can be troublesome when training set is large.

Intuitive for software engineers: k-NN helps them to find the projects that are most similar to the new project.

Learning vs Optimisation

- Predictive models can make mistakes (errors).
- We want to minimise these mistakes.
- The learning algorithm used by some machine learning approaches can be optimisation algorithms whose objective is to minimise some error function.
    - Error function describes how much error a predictive model makes when predicting a set of data.

Image from: http://vignette2.wikia.nocookie.net/fantendo/images/2/25/Mario_Artwork_-_Super_Mario_3D_World.png/revision/latest?cb=20131025223058

Learning vs Optimisation

- However, from the problem point of view, the goal of machine learning is to create models able to generalise to unseen data.
  - In supervised learning: able to make good predictions for new data that were unavailable at training time.
  - We cannot calculate the error on such data during training time!

Image from: http://vignette2.wikia.nocookie.net/fantendo/images/2/25/Mario_Artwork_-_Super_Mario_3D_World.png/revision/latest?cb=20131025223058

Further Reading

https://saravananthirumuruganathan.wordpress.com/2010/05/17/a-detailed-introduction-to-k-nearest-neighbor-knn-algorithm