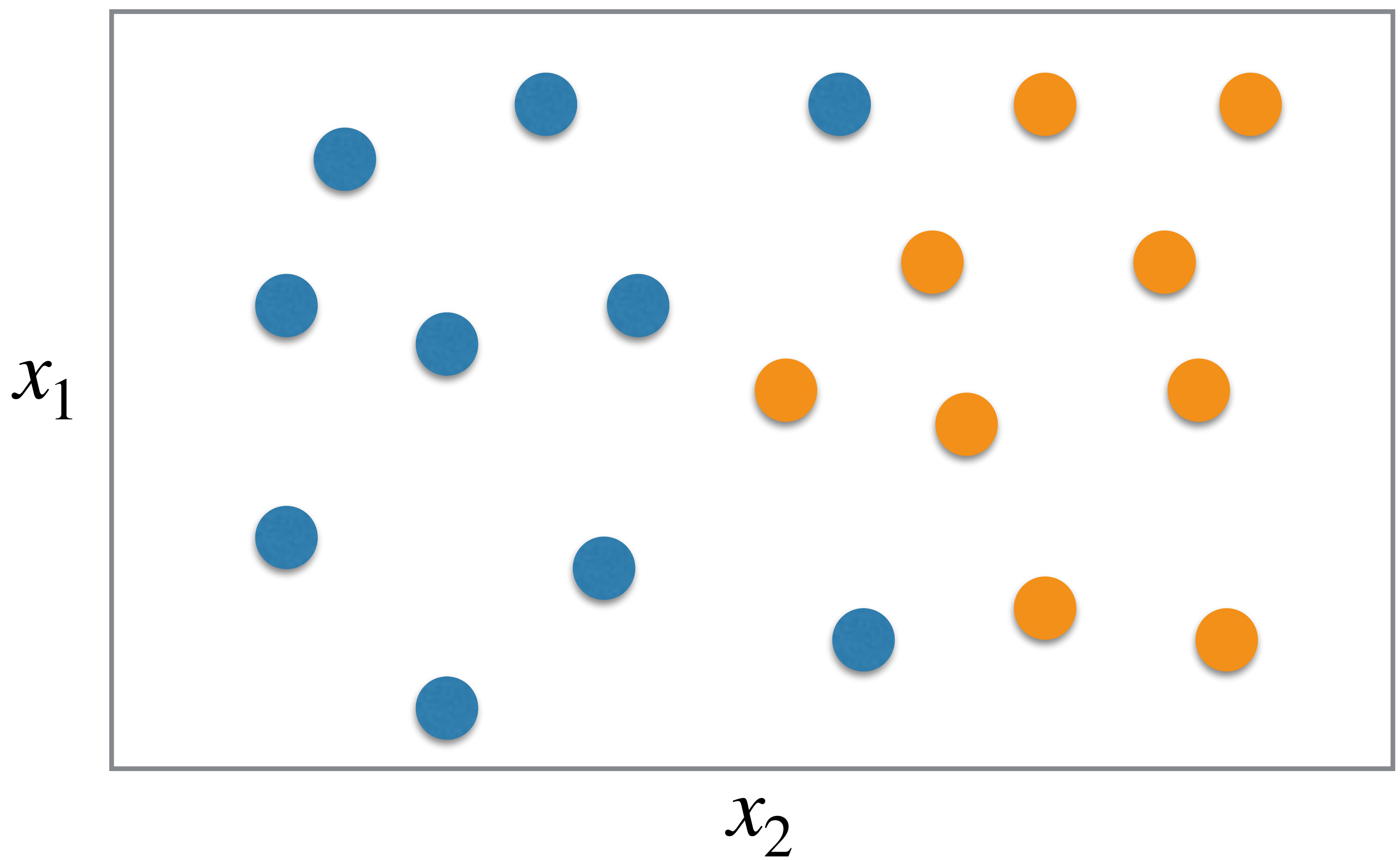
k-Nearest Neighbours

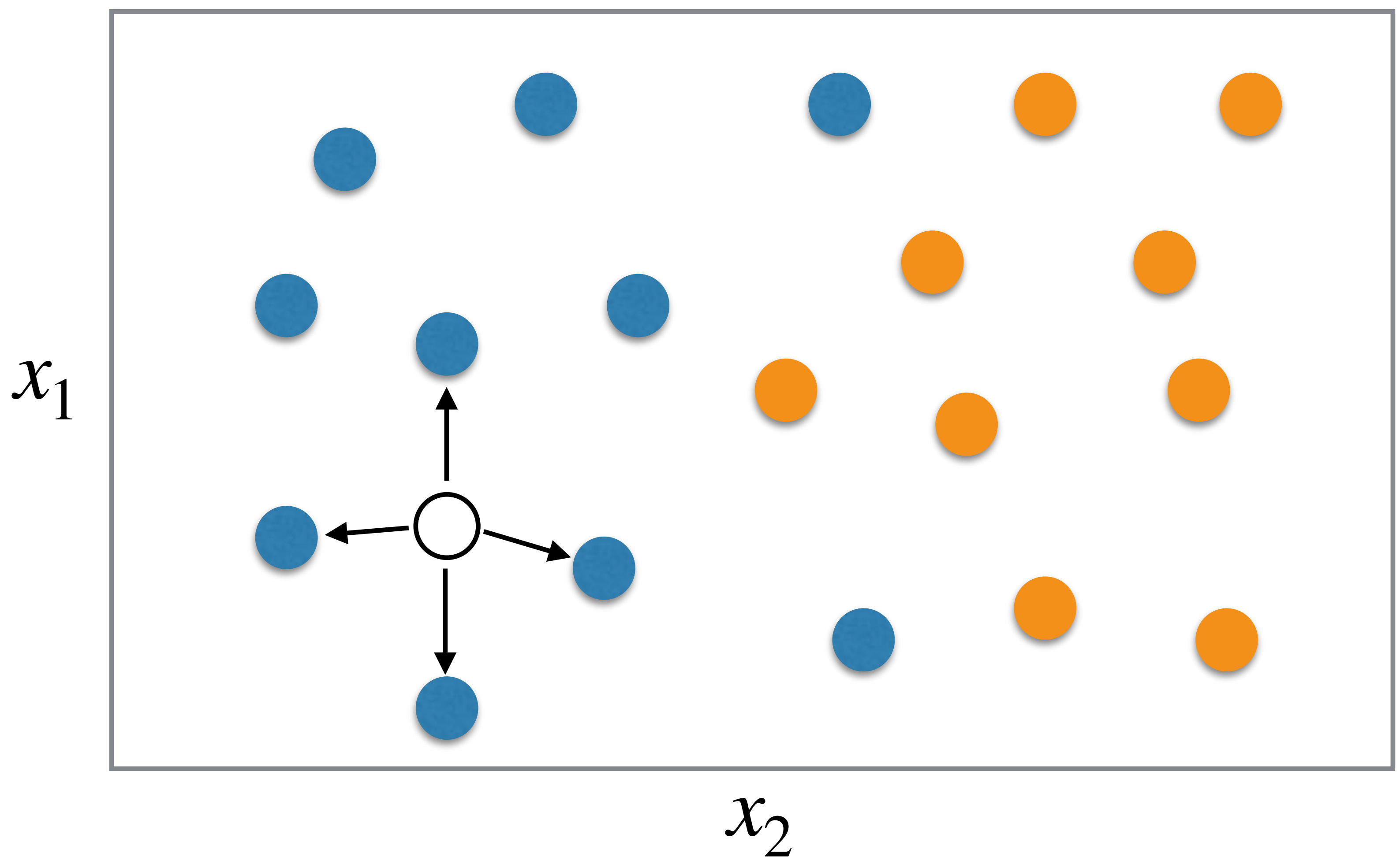
Leandro L. Minku

k-Nearest Neighbours: Basic Idea



$y \in$ {blue, orange}

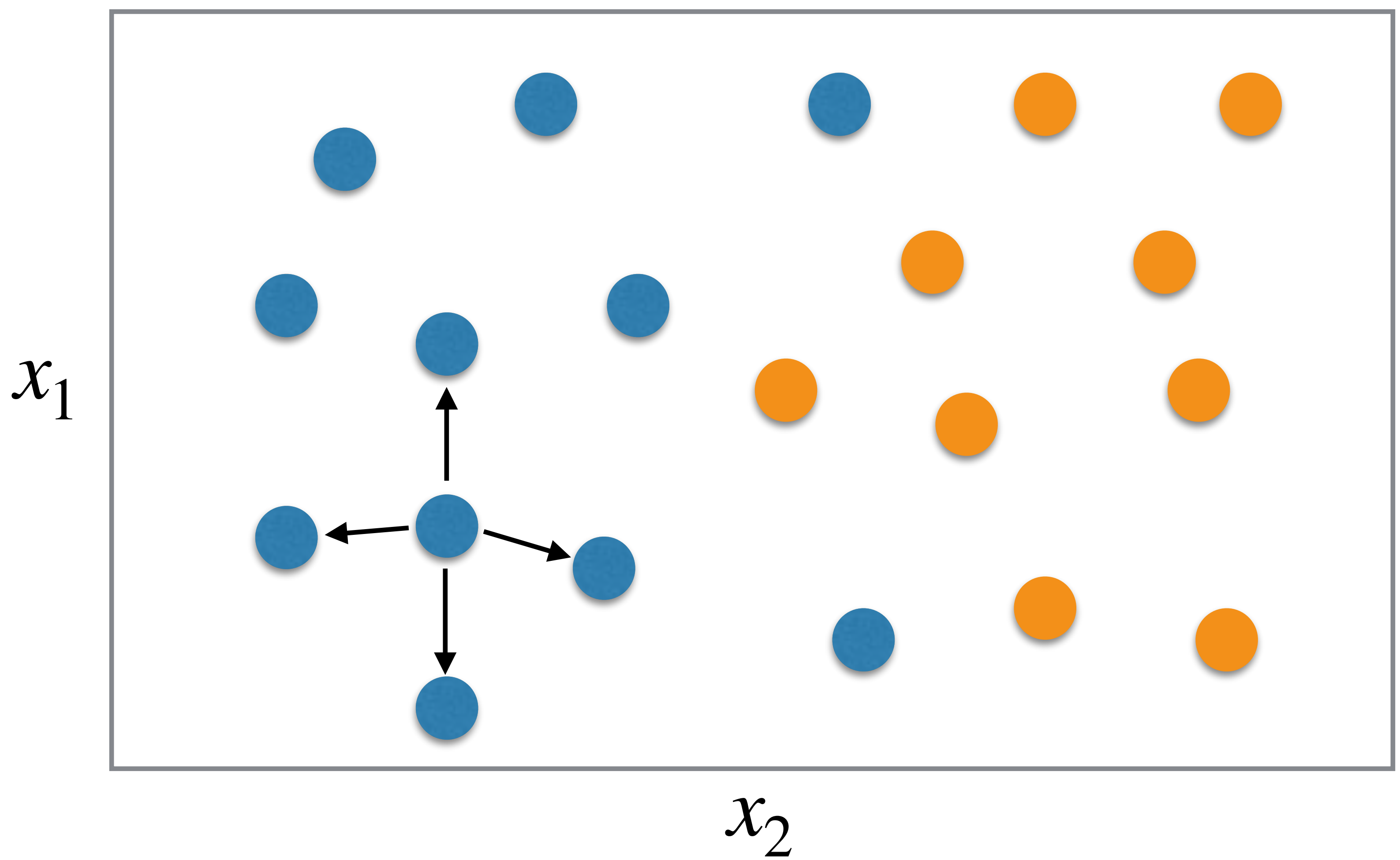
k-Nearest Neighbours: Basic Idea



K = 4

Usually, for classification problems: predict the majority among the values of the dependent variable of the k nearest neighbours (majority vote).

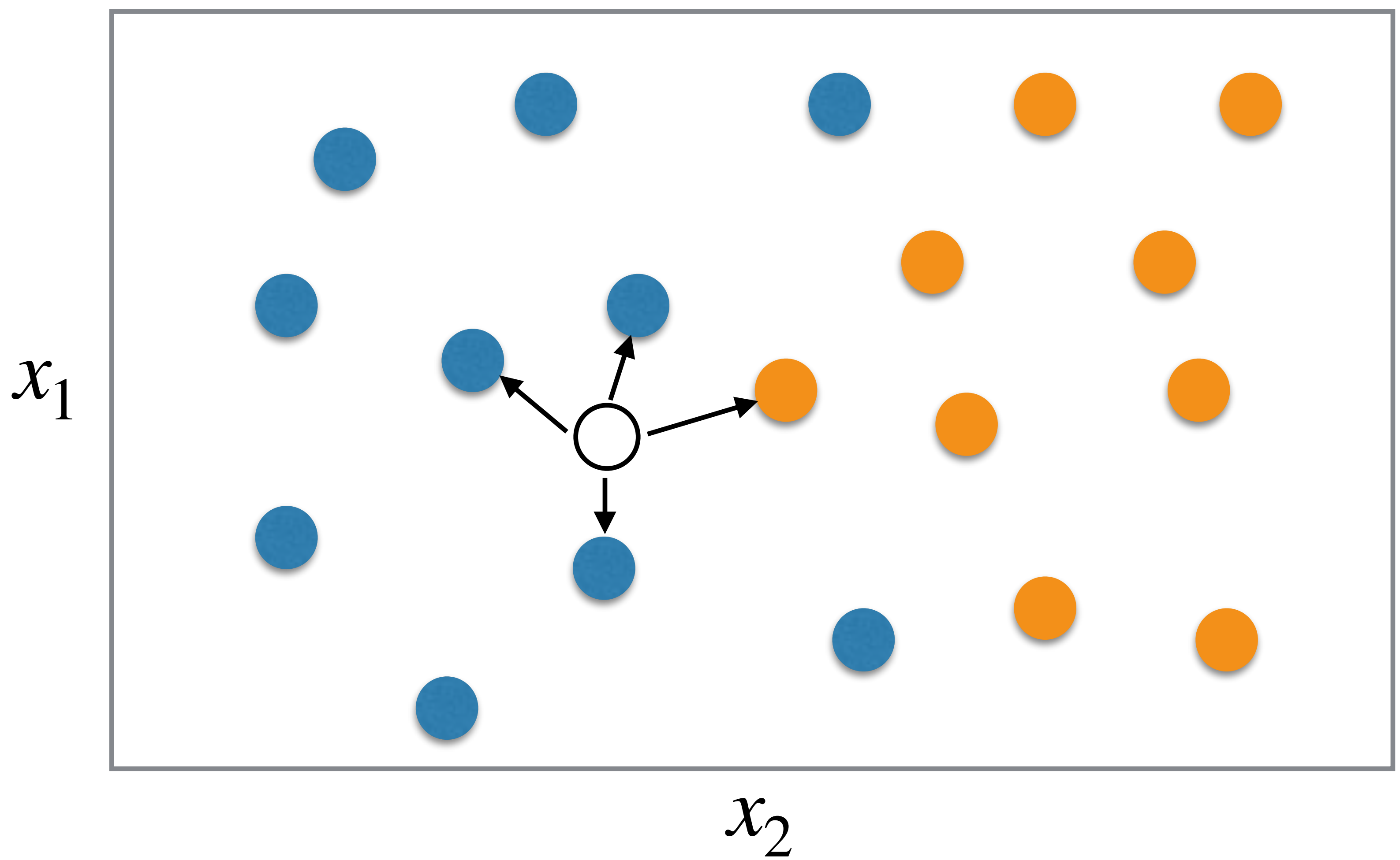
k-Nearest Neighbours: Basic Idea



K = 4

Usually, for classification problems: predict the majority among the values of the dependent variable of the k nearest neighbours (majority vote).

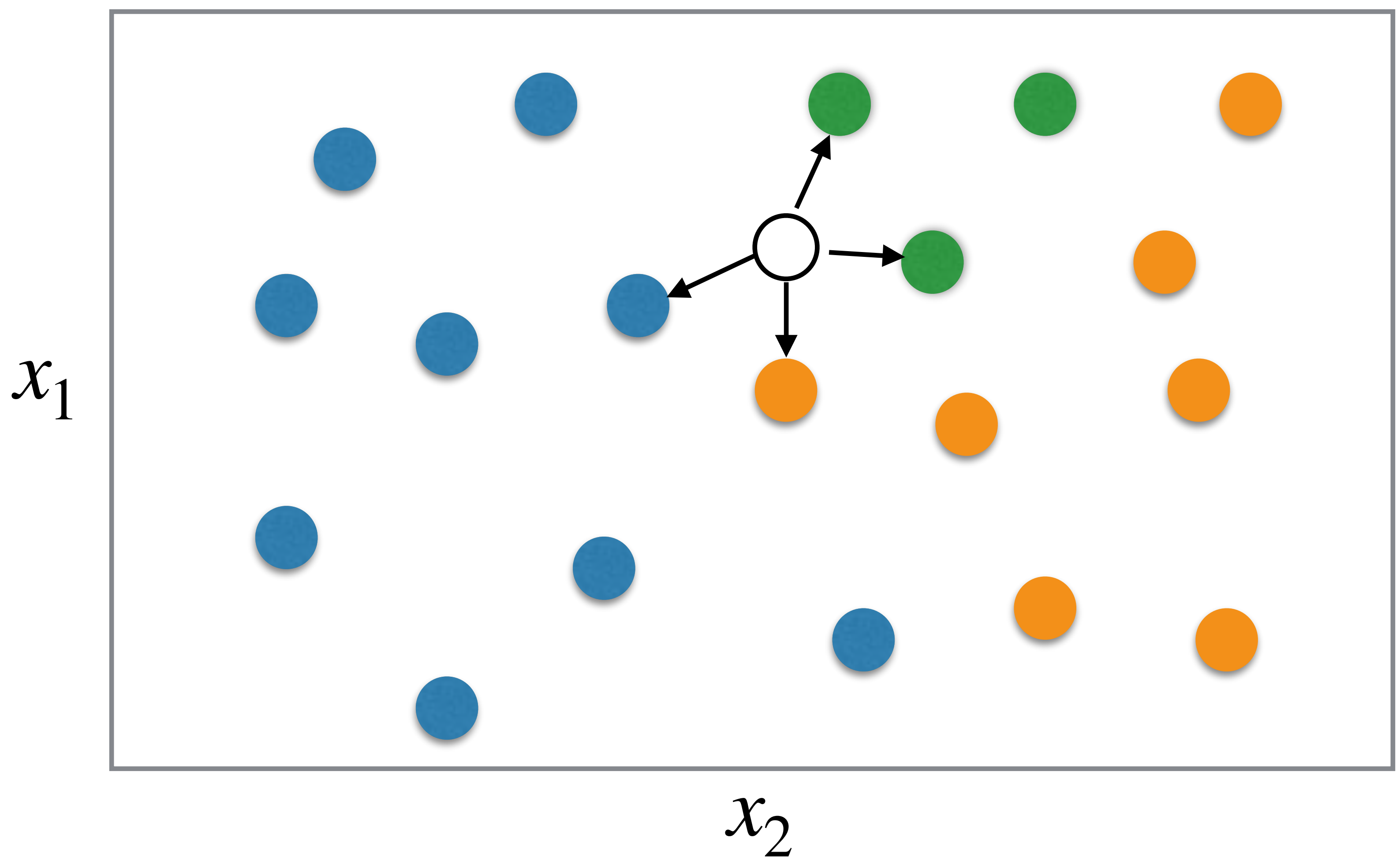
k-Nearest Neighbours: Basic Idea



k = 4

What is the predicted output for this test instance?

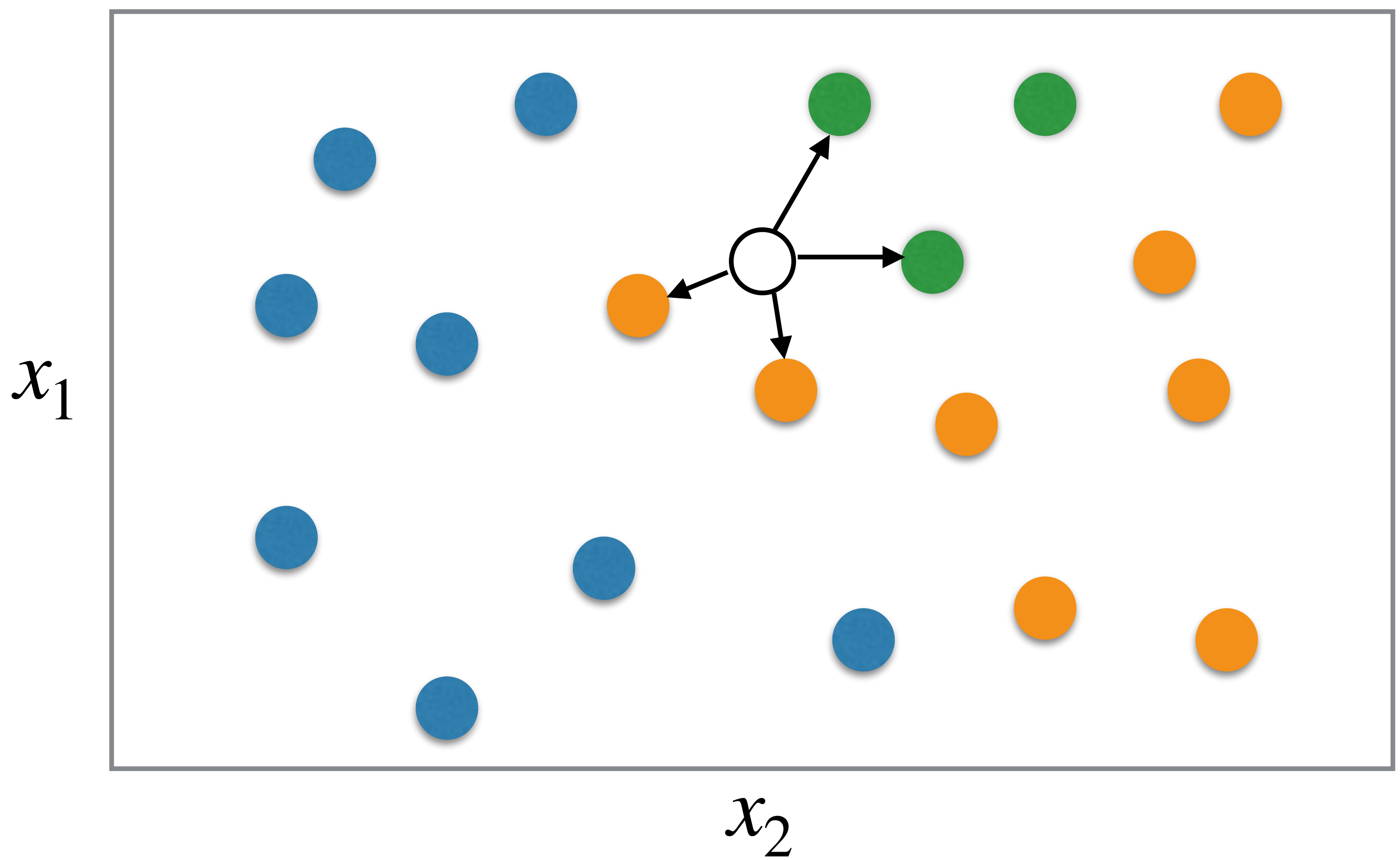
k-Nearest Neighbours: Basic Idea



k = 4

What is the predicted output for this test instance?

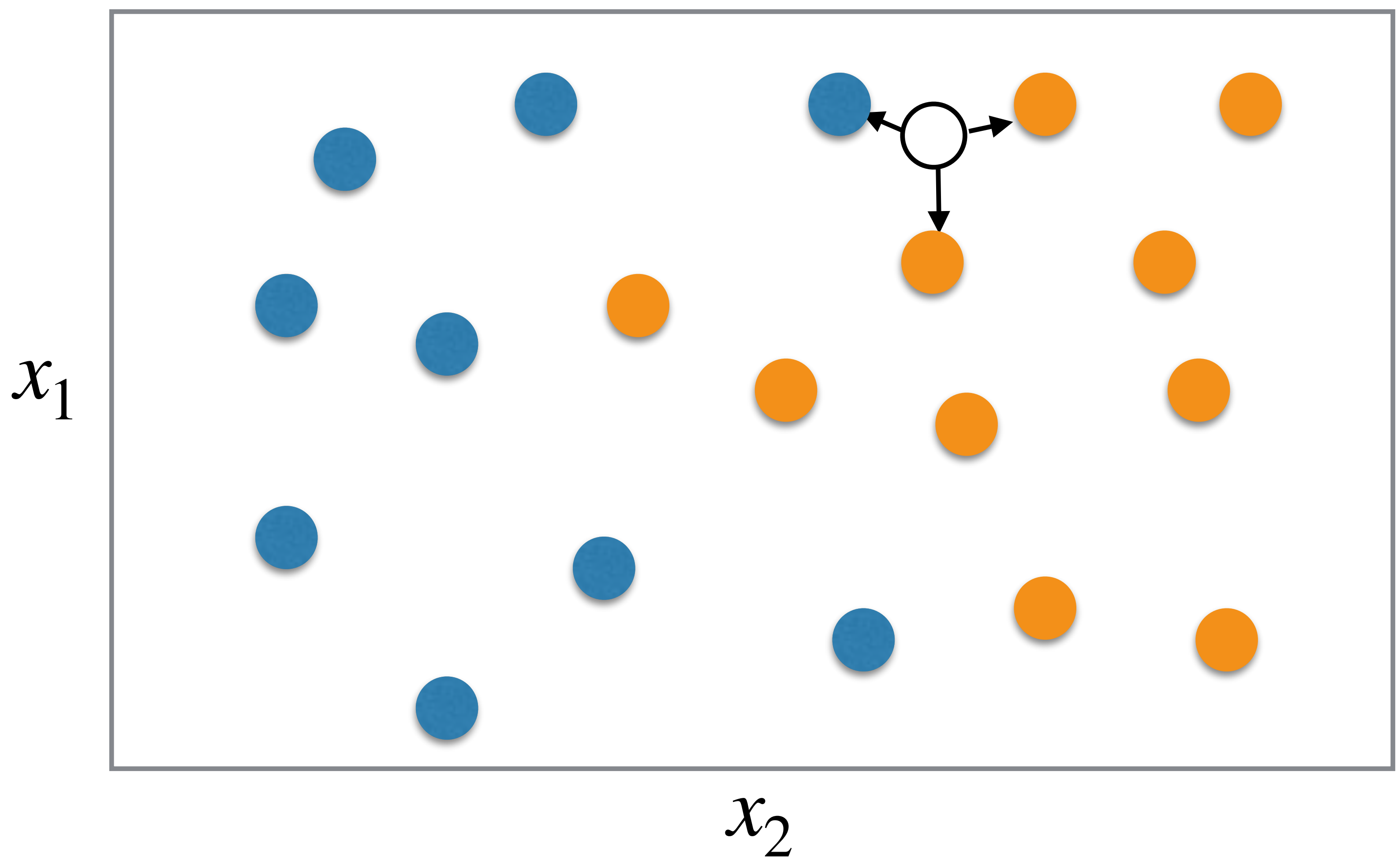
k-Nearest Neighbours: Basic Idea



k = 4

What is the predicted output for this test instance?

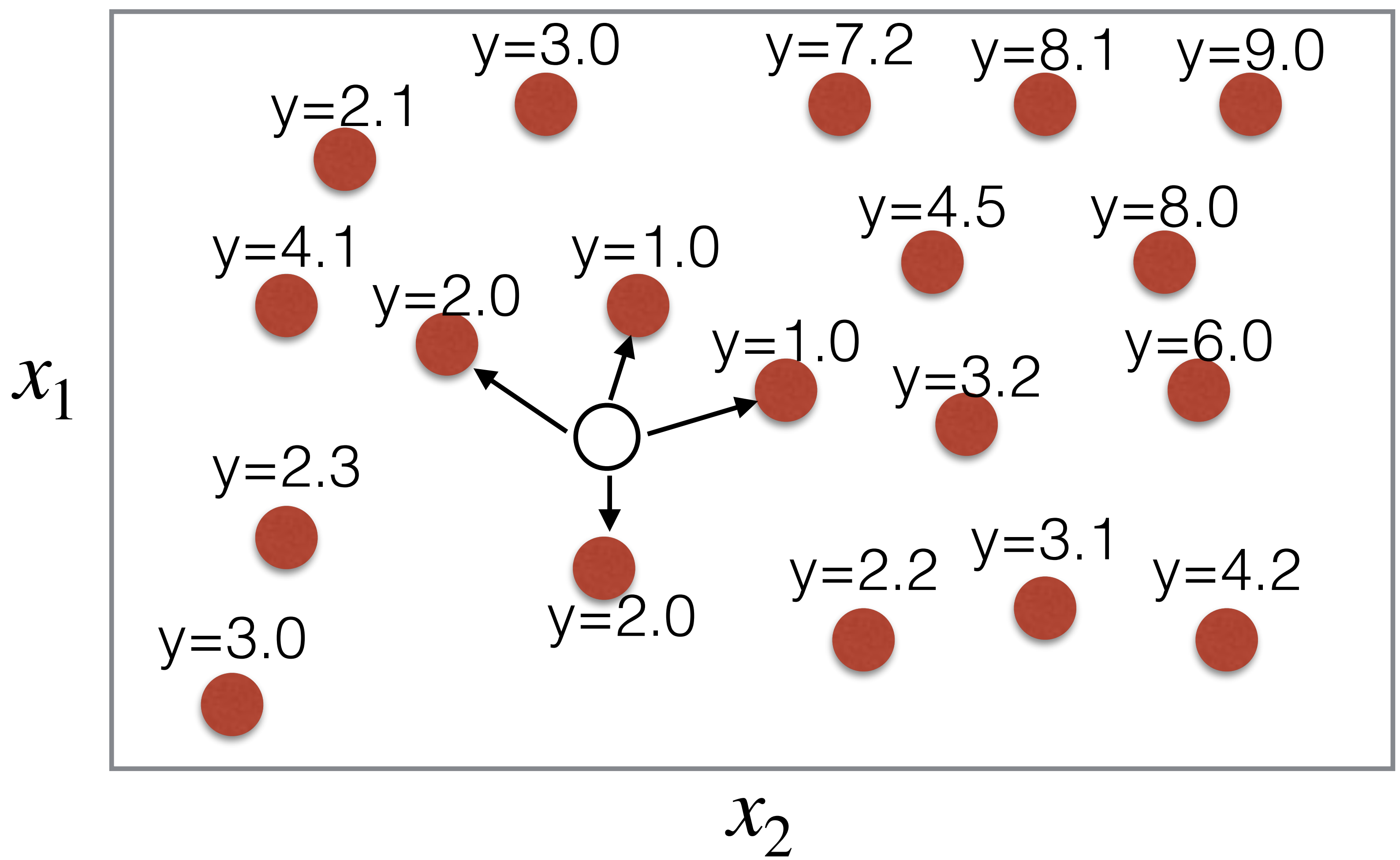
k-Nearest Neighbours: Basic Idea



k = 3

What is the predicted output for this test instance?

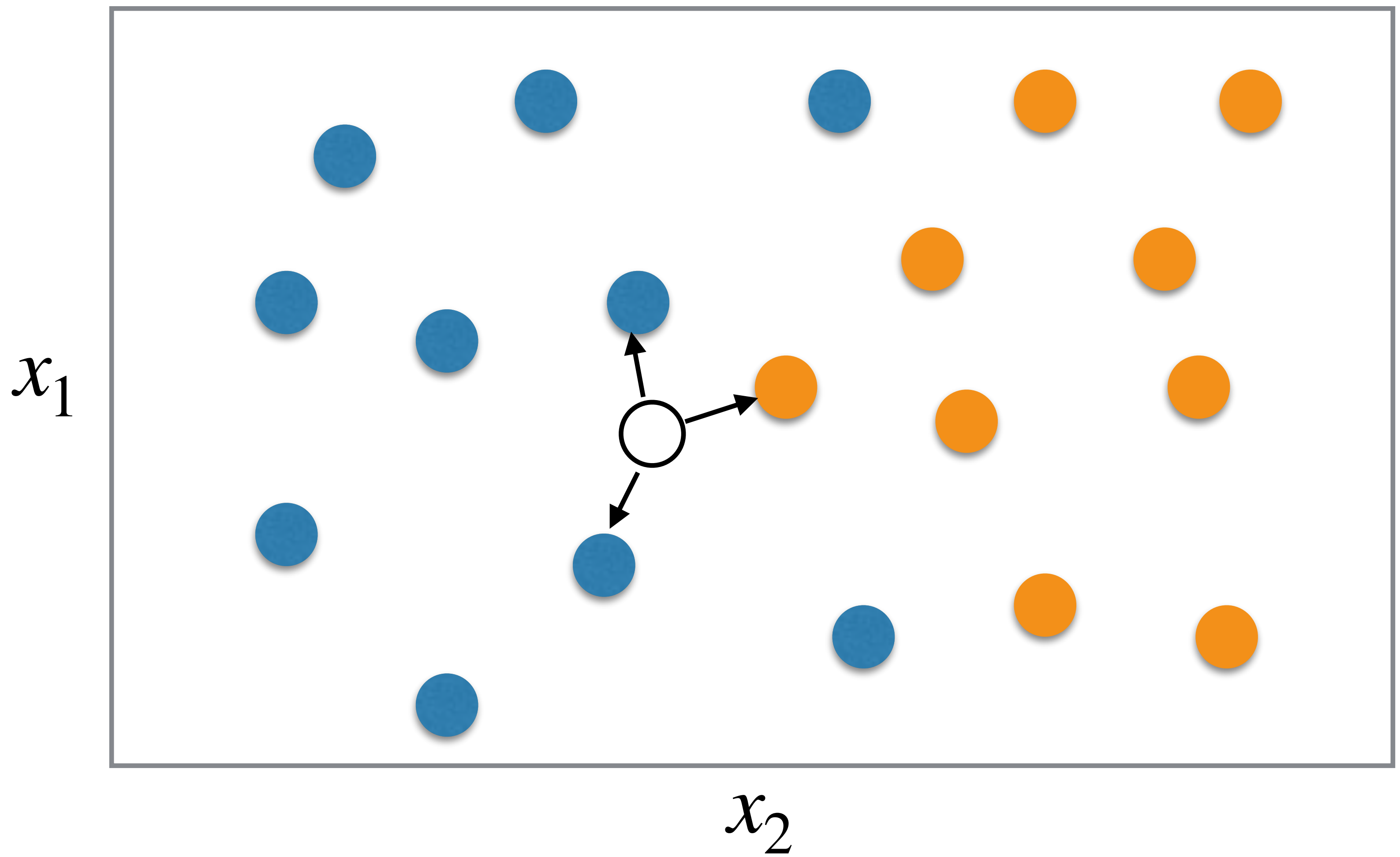
k-Nearest Neighbours: Basic Idea



k = 4

Usually, for regression problems: predict the average among the values of the dependent variable of the k nearest neighbours.

k-Nearest Neighbours: Basic Idea



k = 3

Given an instance to be predicted, we need to find its k nearest neighbours, based on some distance metric **on the input space**.

Distance Metric

- Usually, this is the Euclidean Distance.
- For d dimensions in the input space:

$$\text{distance}(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \sqrt{(x_1^{(i)} - x_1^{(j)})^2 + (x_2^{(i)} - x_2^{(j)})^2 + \cdots + (x_d^{(i)} - x_d^{(j)})^2}$$

$$\text{distance}(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \sqrt{\sum_{p=1}^{d} (x_p^{(i)} - x_p^{(j)})^2} \quad = \sqrt{(\mathbf{x}^{(\mathbf{i})} - \mathbf{x}^{(\mathbf{j})})^T (\mathbf{x}^{(\mathbf{i})} - \mathbf{x}^{(\mathbf{j})})}$$

Normalisation of Numeric Independent Variable

- Problem: different numeric independent variable may have different scales.
  - Scale of numeric independent variable 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 numeric independent variables of all data so that they will be between 0 and 1. E.g.: normalising independent variable *p* of example *i*:

$$\text{normalise}(x_p^{(i)}) = \frac{x_p^{(i)} - \text{min}_p}{\text{max}_p - \text{min}_p}$$

- 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.

Ordinal or Categorical Independent variable

- Independent variable can be numerical, ordinal or categorical.
  - Numeric: 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!

$$\text{distance}(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \sqrt{\sum_{p=1}^{d} (x_p^{(i)} - x_p^{(j)})^2}$$

- For ordinal independent variable, we can convert them to numeric.
  - E.g.: low = 0, medium = 0.5, high = 1.

- For categorical independent variable, we could use the following idea:

$$\text{if } (x_p^{(i)} = x_p^{(j)}),\ (x^{(i)} - x^{(j)}) = 0$$

$$\text{if } (x_p^{(i)} \neq x_p^{(j)}),\ (x_p^{(i)} - x_p^{(j)}) = 1$$

Procedure for Predicting a New (Test) Example

- Given the independent variables of a new example $(\mathbf{x}^{(i)},?)$, the number of neighbours k, and **min** and **max** observed so far.

- Update **min** and **max** based on $\mathbf{x}^{(i)}$

- For each training example $(\mathbf{x}^{(j)},y^{(j)})$
    - dist = distance(normaliseEachVar($\mathbf{x}^{(i)}$,**min**,**max**),normaliseEachVar($\mathbf{x}^{(j)}$,**min**,**max**))
    - Add (dist, $y^{(j)}$) to a data structure T sorted based on ascending order of distance.

- Return the majority vote (or average) of $y^{(j)}$ for the first k entries of T.

k-NN Approach

- k-NN Learning Algorithm:
  - No real training; simply store all training data received so far, together with the maximum and minimum values of the numerical independent variables.
- k-NN "Model":
  - All training data received so far, together with the maximum and minimum values of the numerical independent variables.
- k-NN prediction for an instance ($\mathbf{x}^{(i)}$,?):
  - Find the K nearest neighbours, i.e., the K training examples that are the closest to $\mathbf{x}^{(i)}$.
  - For classification problems: majority vote.
  - For regression problems: average.

Advantages and Disadvantages

- Advantages:
  - Training is simple and quick: store the training data.
- Disadvantage:
  - Memory requirements are high: stores all data, which can be troublesome when training set is very 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 very large.

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

Advantages and Disadvantages

- Advantages:
    - Training is simple and quick: store the training data.
- Disadvantage:
    - Memory requirements are high: stores all data, which can be troublesome when training set is very 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 very large.

k-NN can be good for applications where there is little data.

Advantages and Disadvantages

- Advantages:
  - Training is simple and quick: store the training data.
- Disadvantage:
  - Memory requirements are high: stores all data, which can be troublesome when training set is very 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 very large.

Intuitive: k-NN helps people to find the examples that are most similar to the new example.

Quiz

- Consider that you have an example with $\mathbf{x}^{(1)} = (5,10)^T$. Consider also that $min_1 = 0$, $max_1 = 10$, $min_2 = 0$, $max_2 = 20$.
  What is the normalised value of $\mathbf{x}^{(1)}$?

- Consider a given training set containing the following normalised examples:
  - $\mathbf{x}^{(1)}=(0.1,0.1)^T$, $y^{(1)}$ = red
  - $\mathbf{x}^{(2)}=(0.1,0.2)^T$, $y^{(2)}$ = blue
  - $\mathbf{x}^{(3)}=(0.2,0.2)^T$, $y^{(3)}$ = green

  What class would be predicted for a normalised test example $\mathbf{x}^{(4)}=(0.3,0.3)^T$ when k=1?

Further Reading

- Essential:
  - Iain Style's notes on "Classification and k-Nearest Neighbours".
- Recommended:
  - Section 2.5.2 of Bishop, Pattern Recognition and Machine Learning, contains a brief treatment of nearest-neighbour methods.
  - A Detailed Introduction to K-Nearest Neighbor (KNN) Algorithm by Saravanan Thirumuruganathan. Access at: https://saravananthirumuruganathan.wordpress.com/2010/05/17/a-detailed-introduction-to-k-nearest-neighbor-knn-algorithm
- Suggested:
  - Russell and Norvig's "Artificial Intelligence: A Modern Approach"
    - Section 18 (Learning from Examples) up to the end of section 18.2 (Supervised Learning).
    - Section 18.8 (Non-Parametric Models) up to the end of section 18.8.1 (Nearest Neighbour Models).