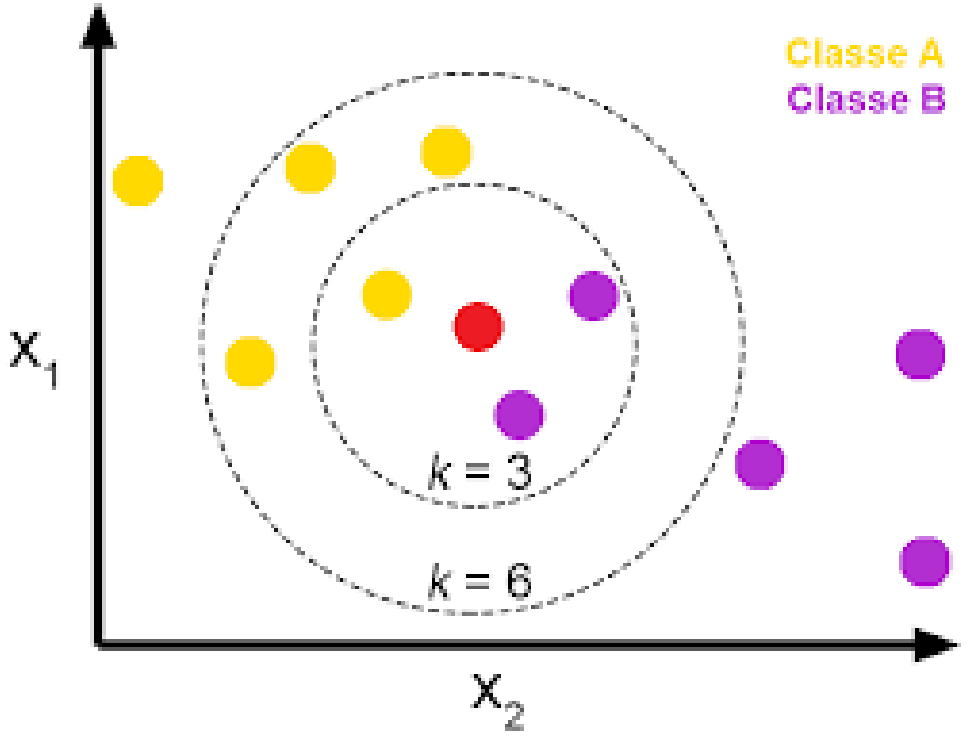
K NEAREST NEIGHBORS

- Nguyễn Hoàng Yến Như
- Nguyễn Trần Phúc Nghi
- Nguyễn Trần Phúc An
- Nguyễn Đức Anh Phúc
- Trịnh Thị Thanh Trúc
- KS. Cao Bá Kiệt
- KS. Quan Chí Khánh An
- KS. Lê Ngọc Huy
- CN. Bùi Cao Doanh
- CN. Nguyễn Trọng Thuận
- KS. Phan Vĩnh Long
- KS. Nguyễn Cường Phát
- ThS. Nguyễn Hoàng Ngân
- KS. Hồ Thái Ngọc
- ThS. Đỗ Văn Tiến
- ThS. Nguyễn Hoàn Mỹ
- ThS. Dương Phi Long
- ThS. Trương Quốc Dũng
- ThS. Nguyễn Thành Hiệp
- ThS. Nguyễn Võ Đăng Khoa
- ThS. Võ Duy Nguyên
- TS. Nguyễn Văn Tâm
- ThS. Trần Việt Thu Phương
- TS. Nguyễn Tấn Trần Minh Khang

NEAREST NEIGHBOR CLASSIFIER

Nearest Neighbor Classifier

– Assign label of nearest training data point to each test data point.



Nearest Neighbor Classifier

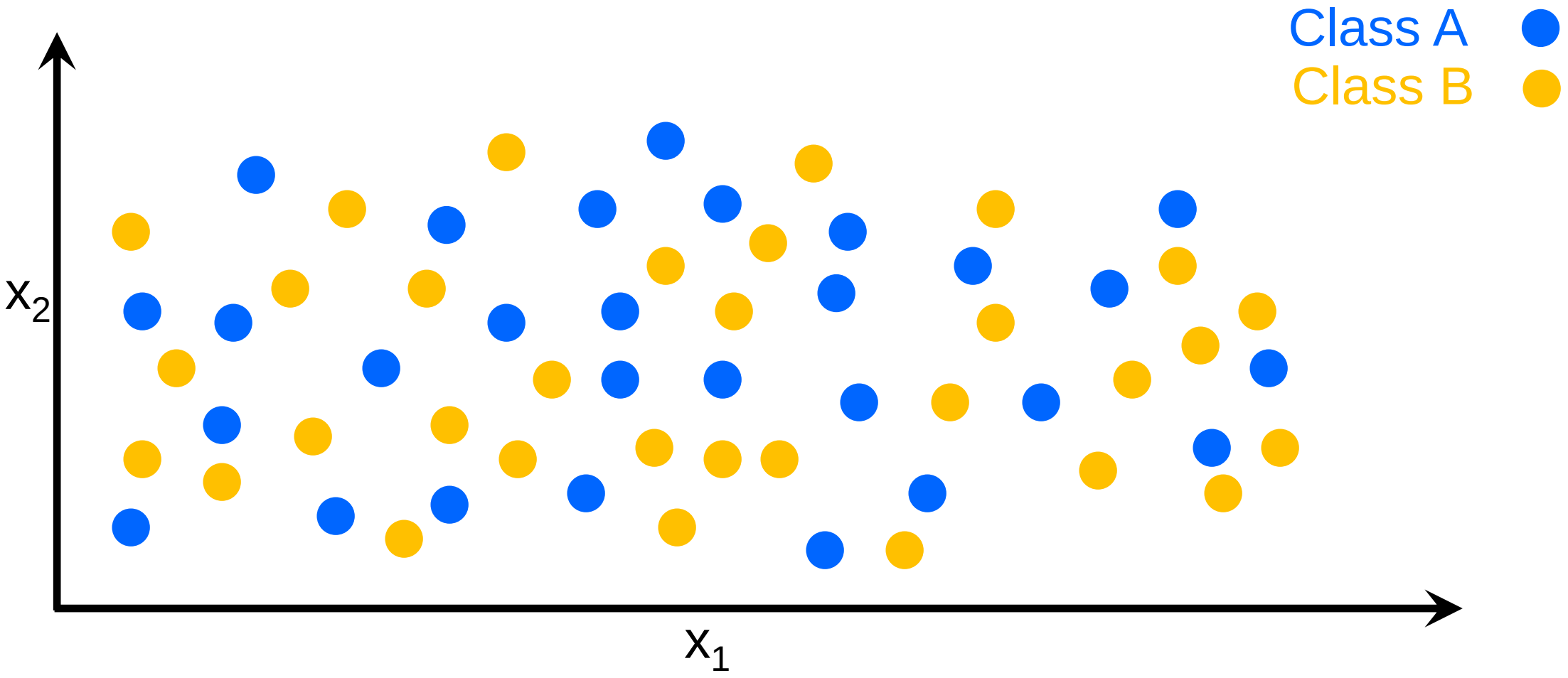
f(**x**) = label of the training example nearest to **x**

–All we need is a distance function for our inputs.

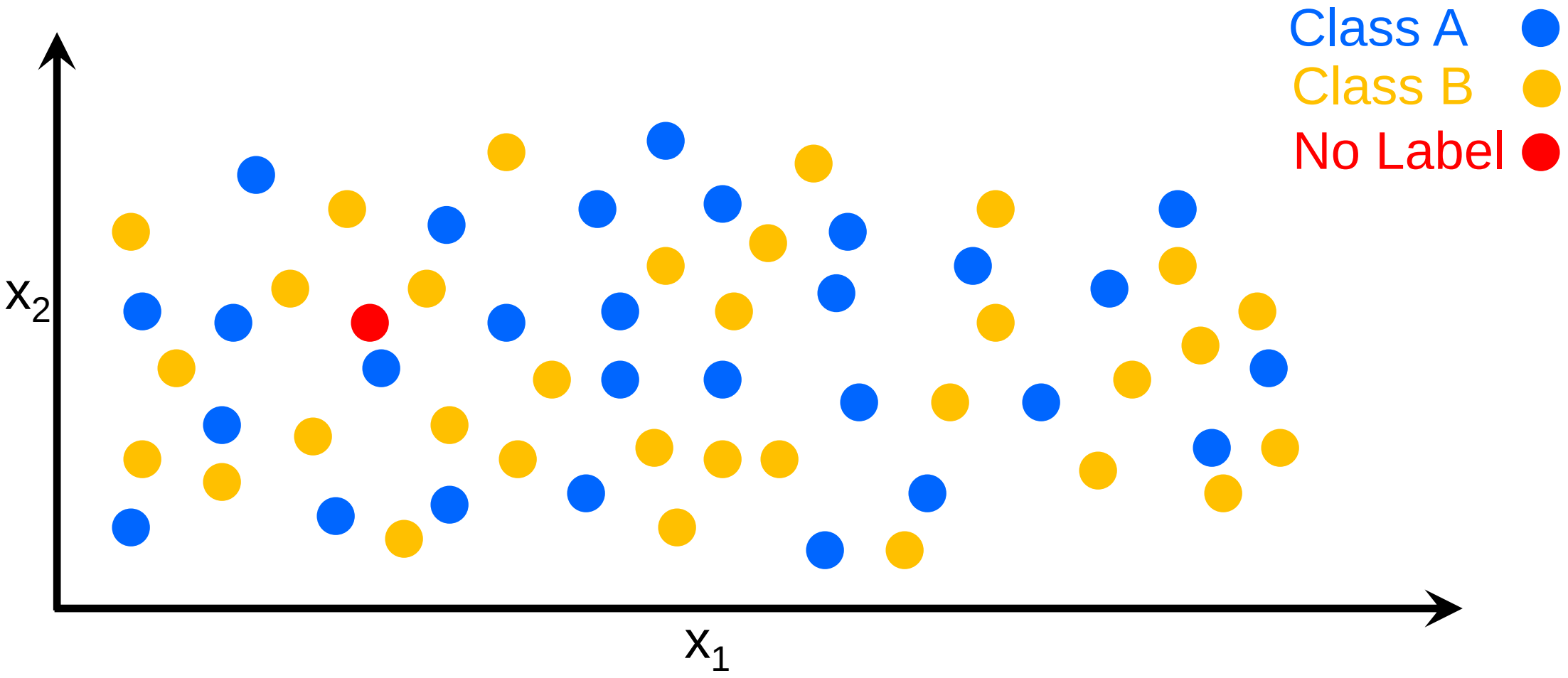
–No training required!

EXAMPLE K-NN

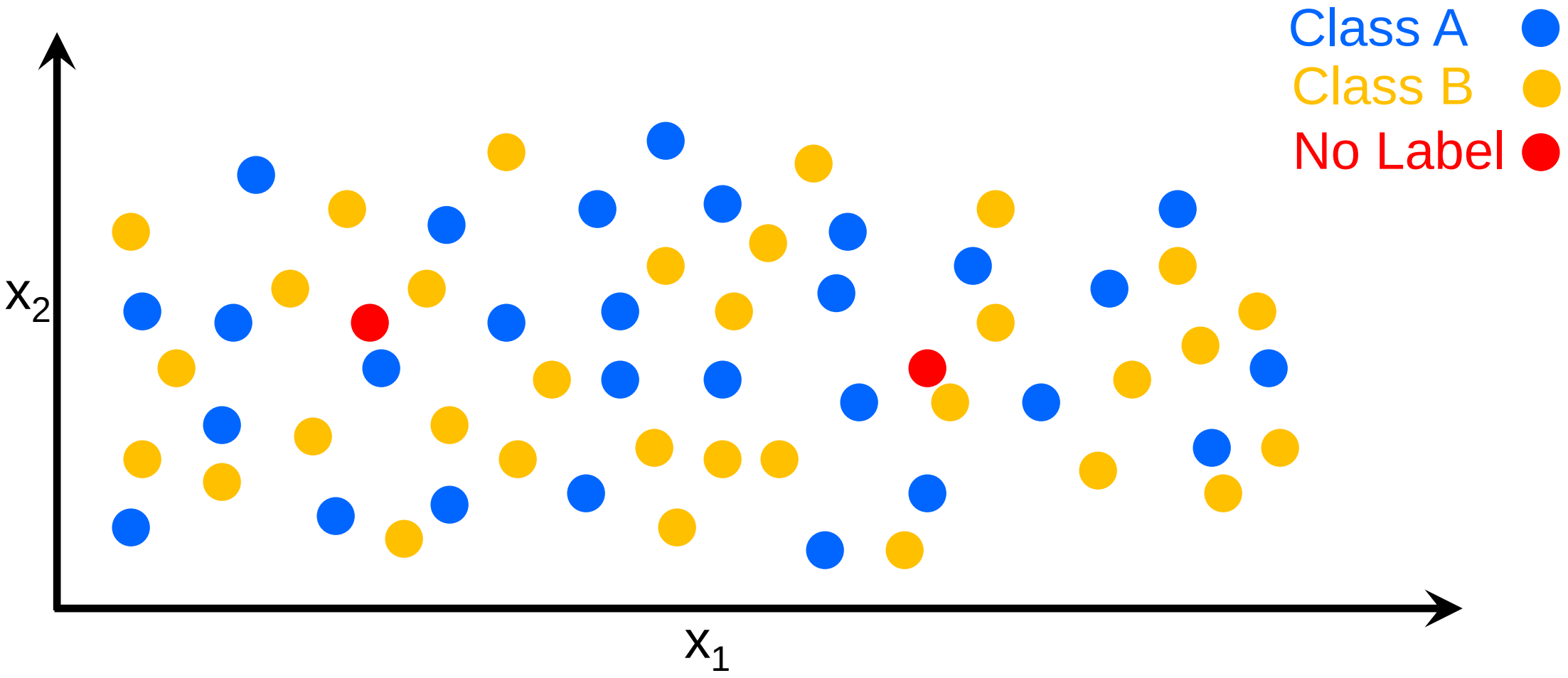
Nearest Neighbor Classifier



Nearest Neighbor Classifier

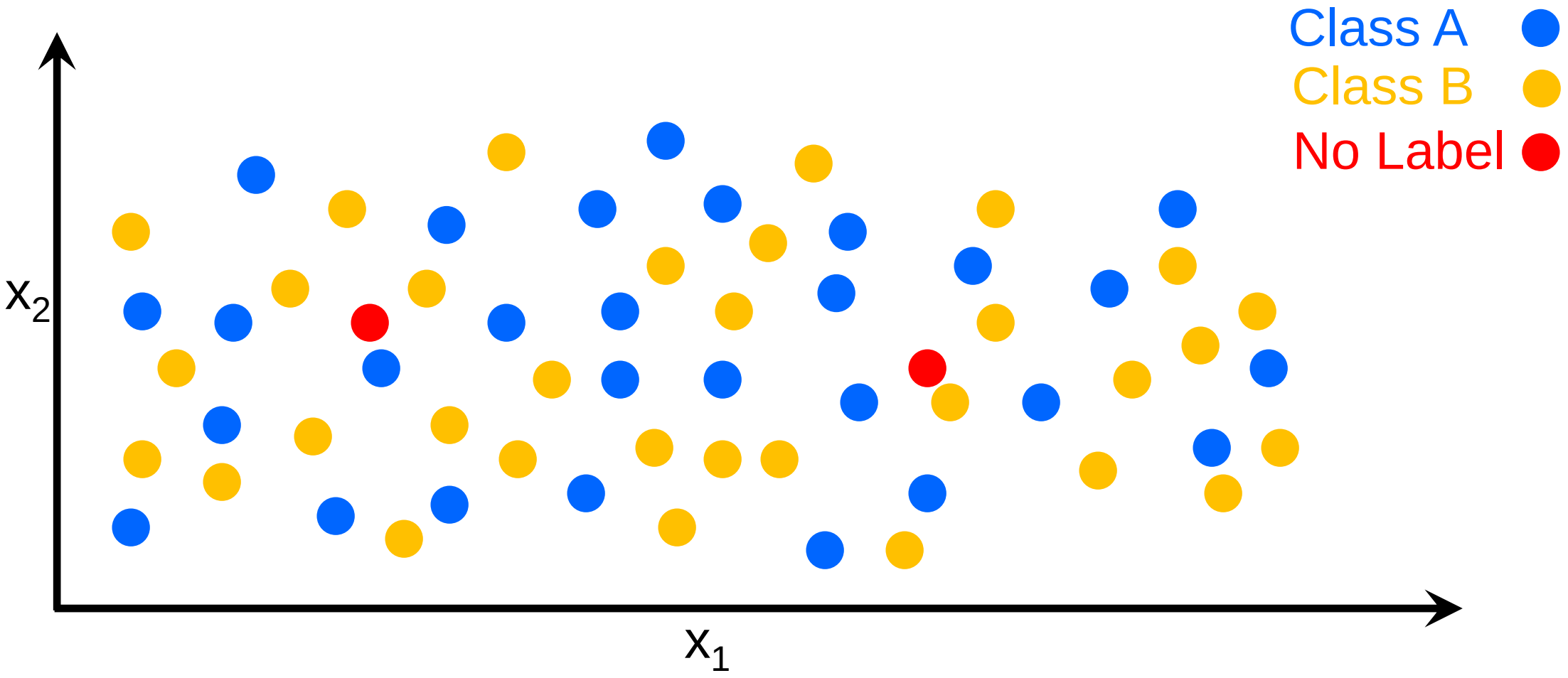


Nearest Neighbor Classifier

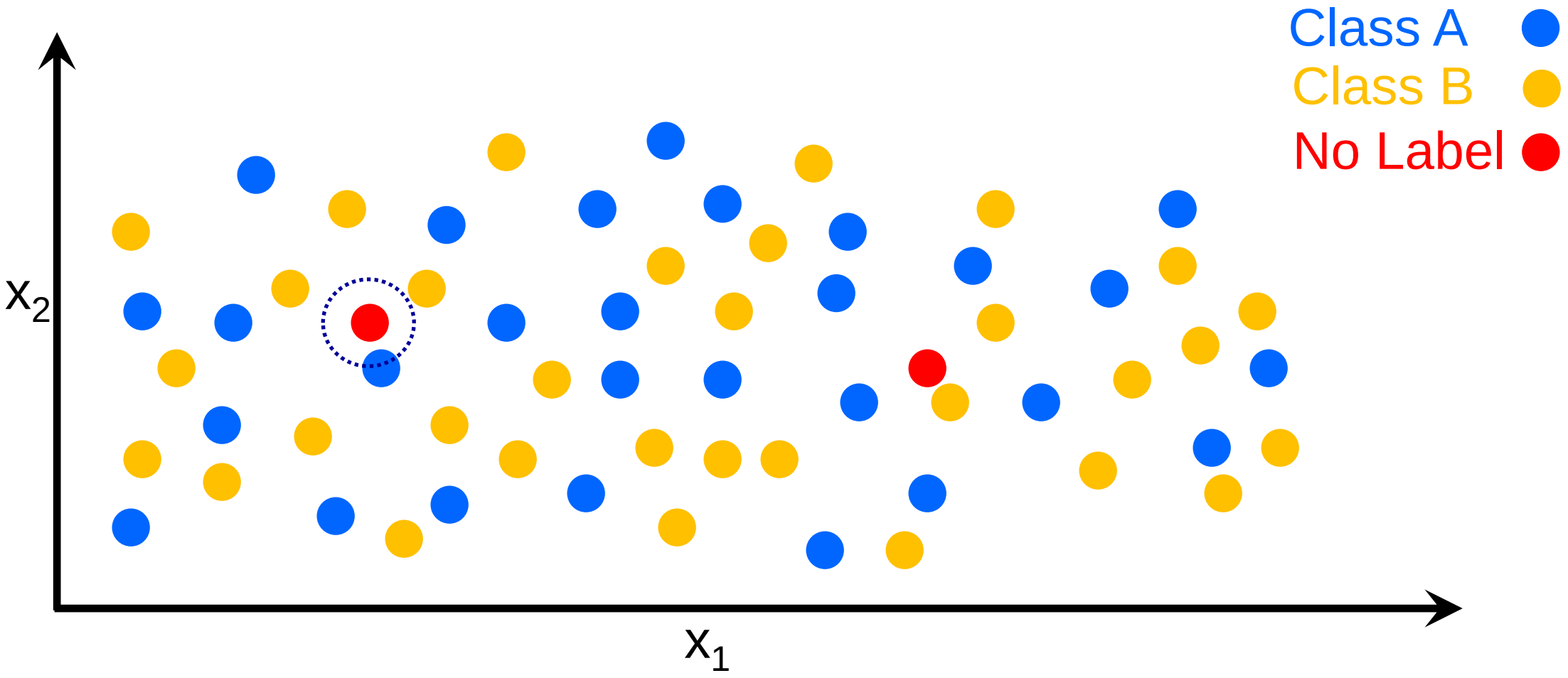


1 – NEAREST NEIGHBOR

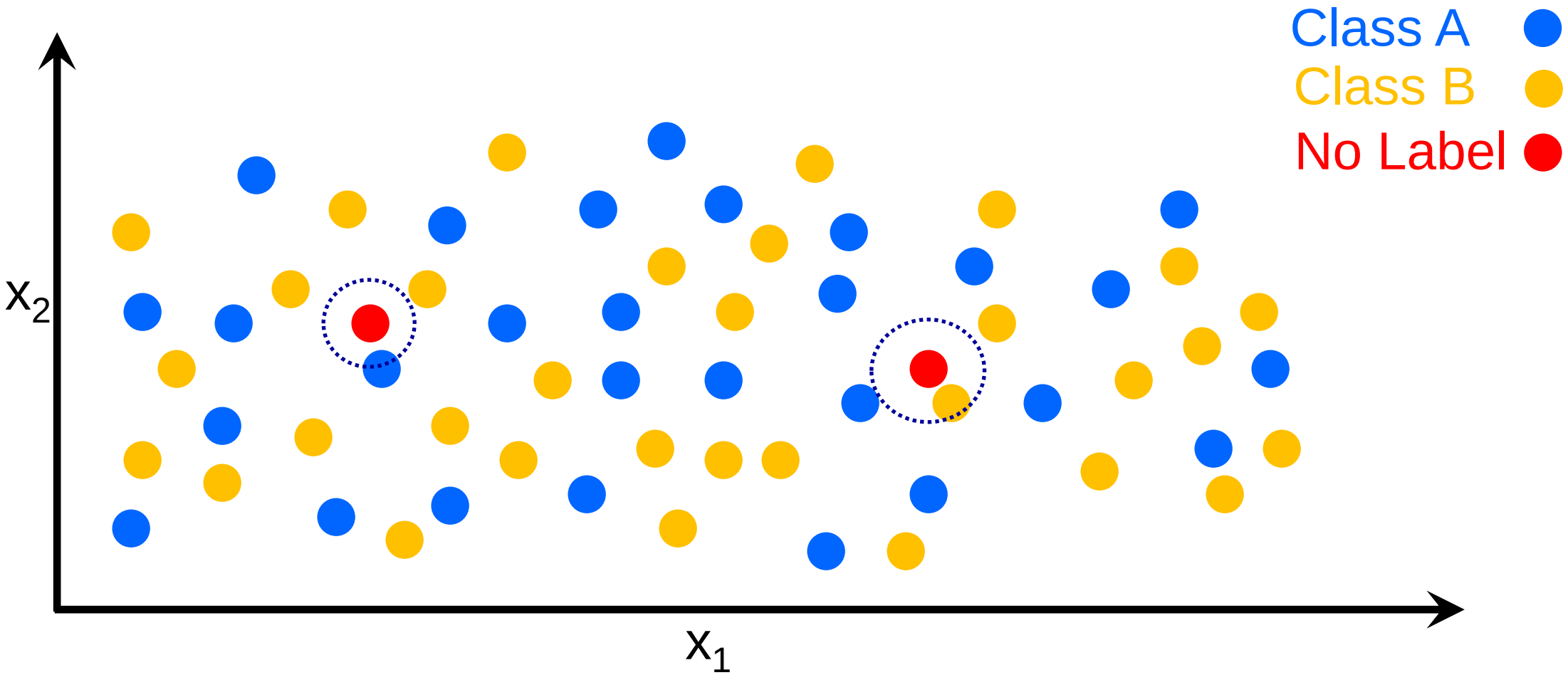
1 – Nearest neighbor



1 – Nearest neighbor

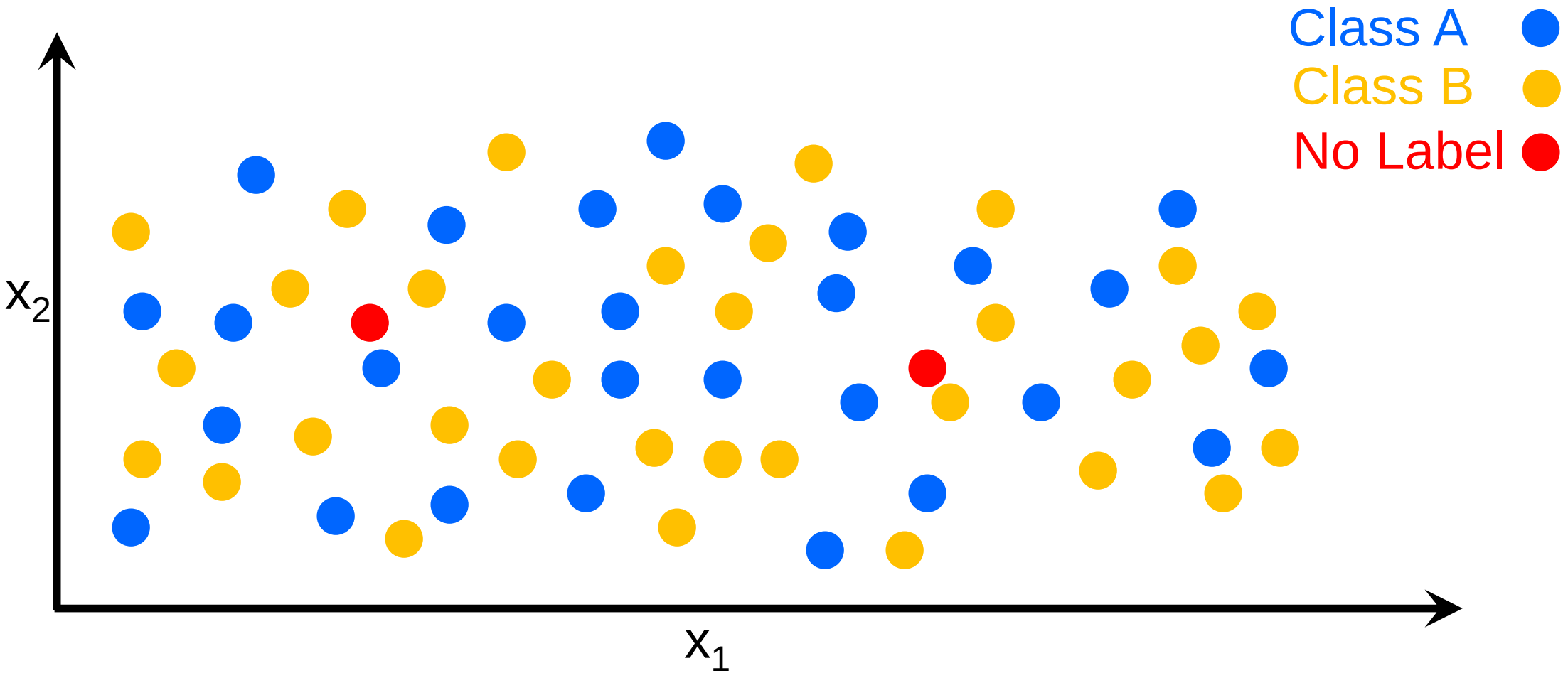


1 – Nearest neighbor

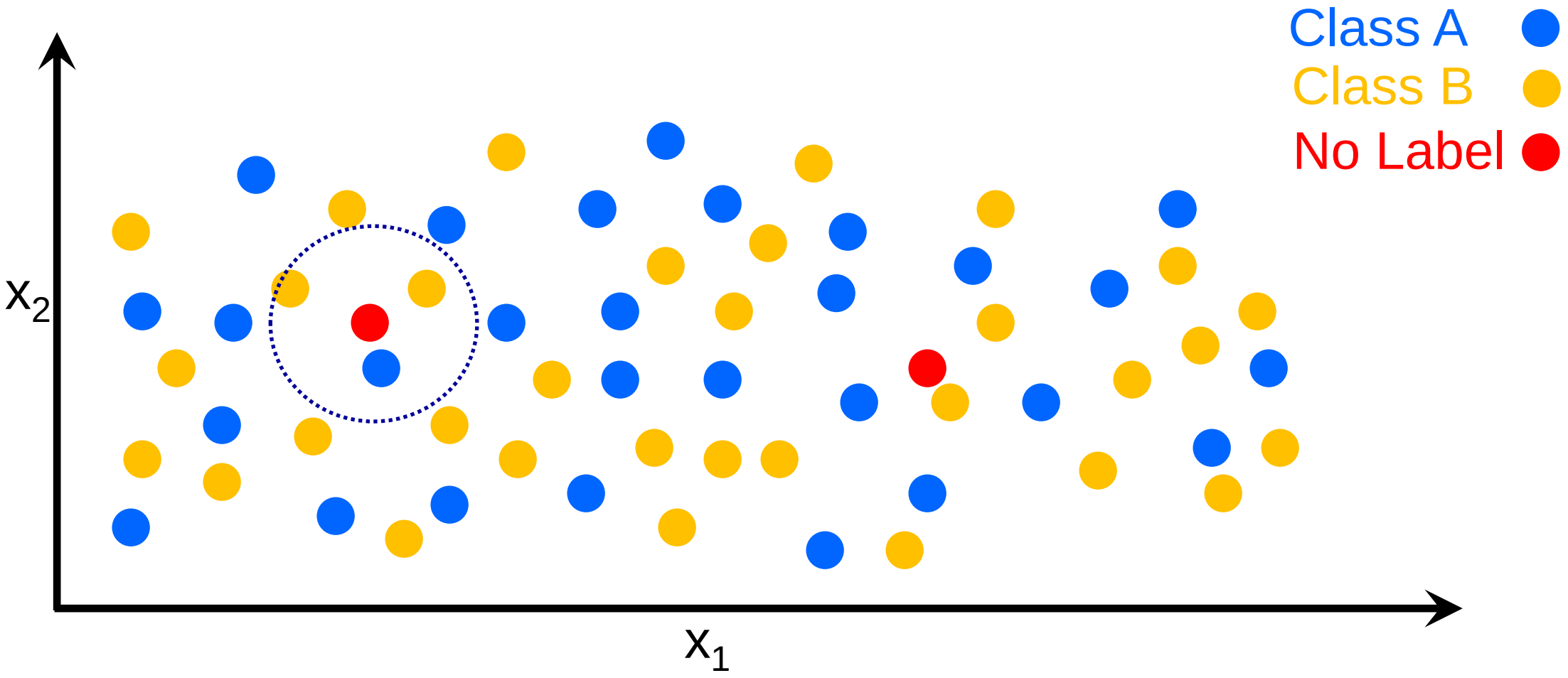


3 – NEAREST NEIGHBOR

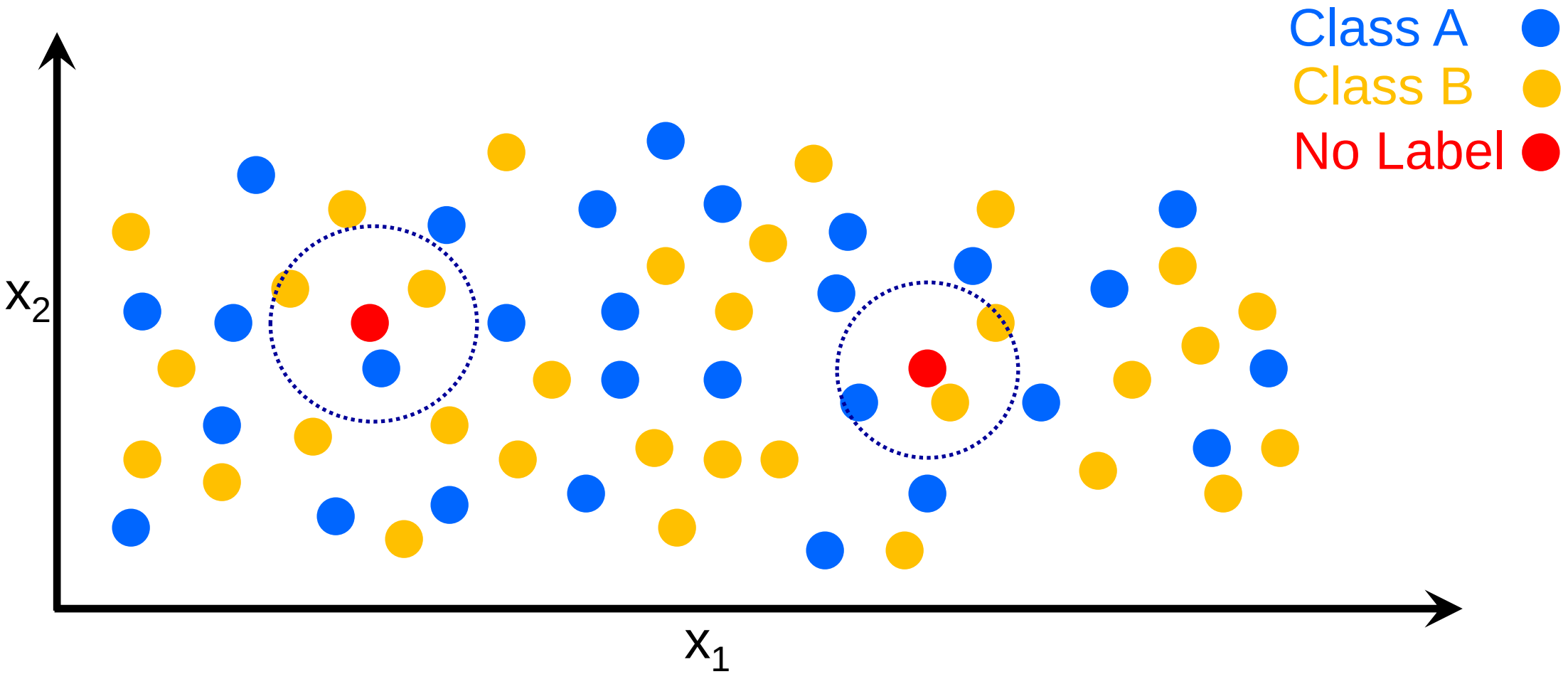
3 – Nearest neighbor



3 – Nearest neighbor

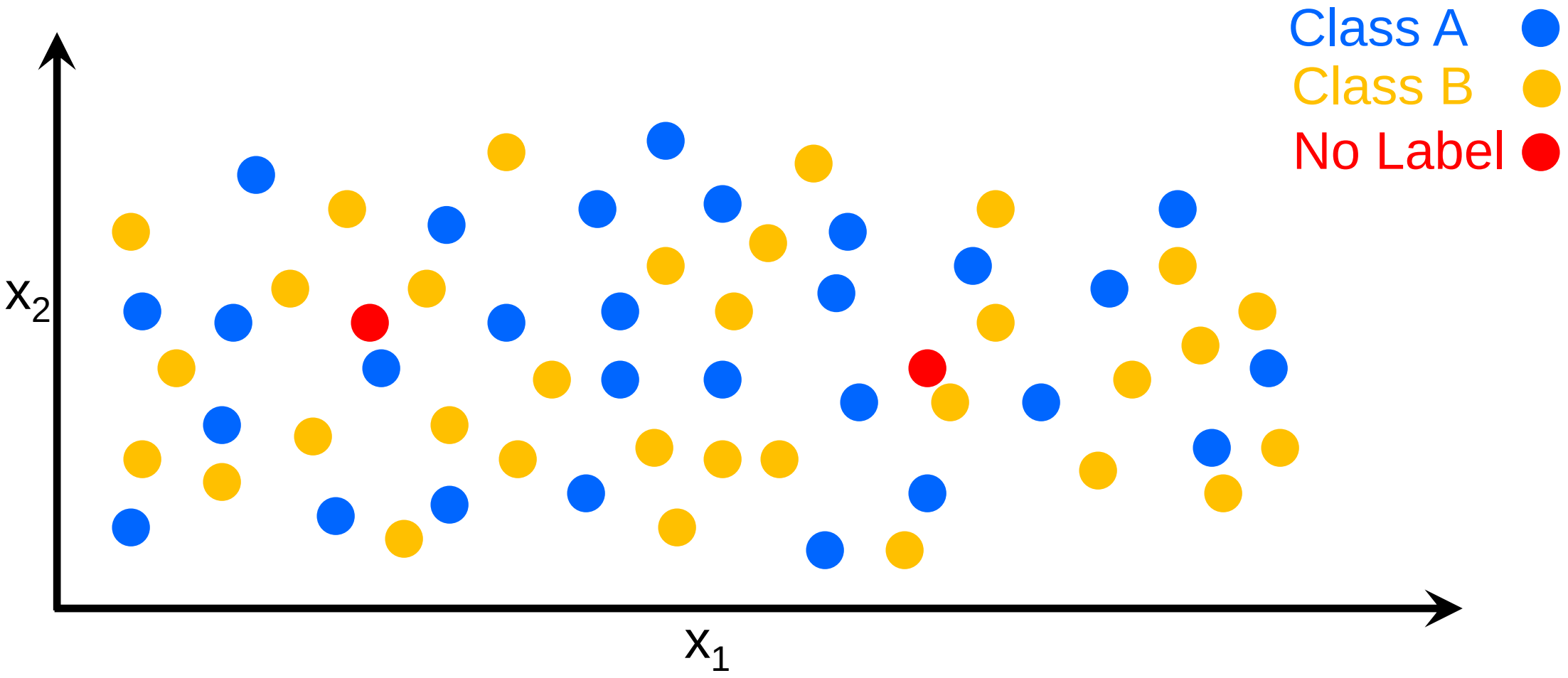


3 – Nearest neighbor

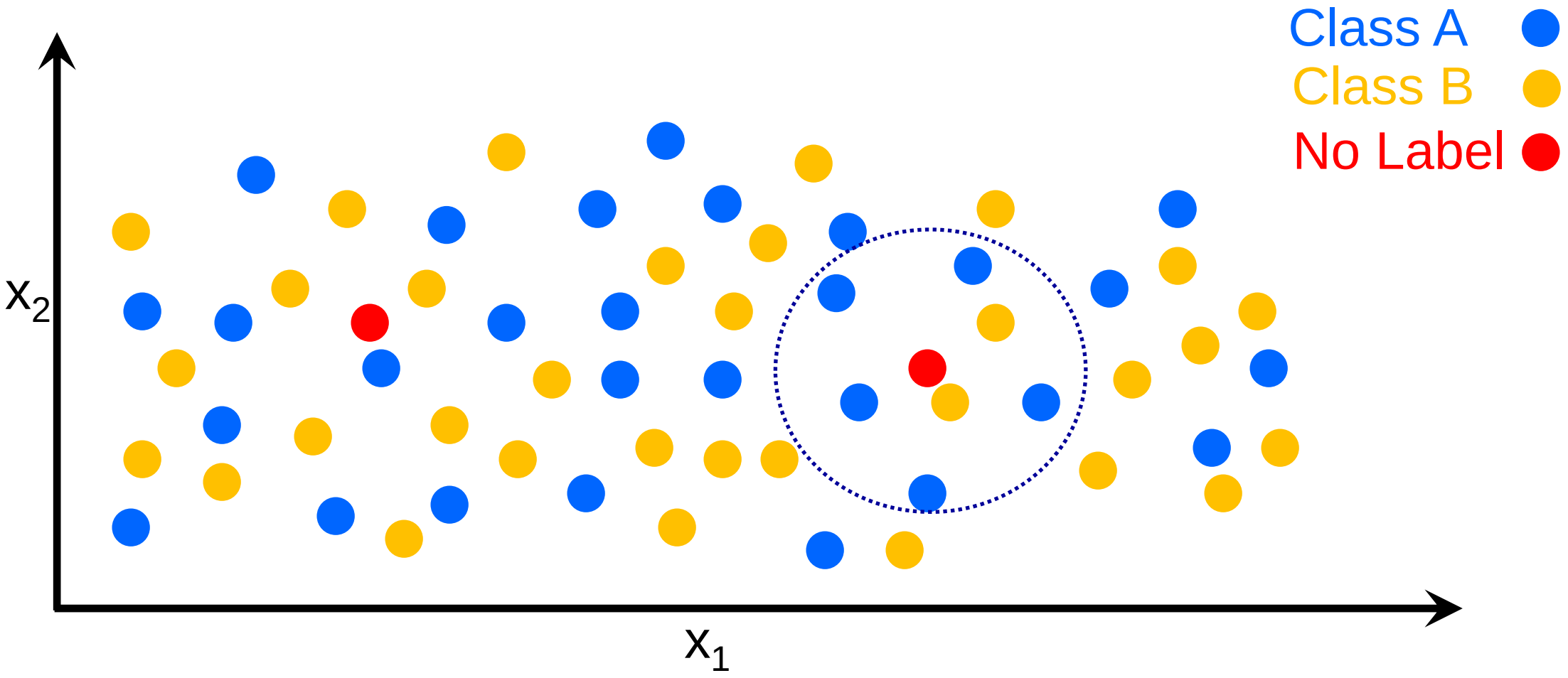


7 – NEAREST NEIGHBOR

7 – Nearest neighbor

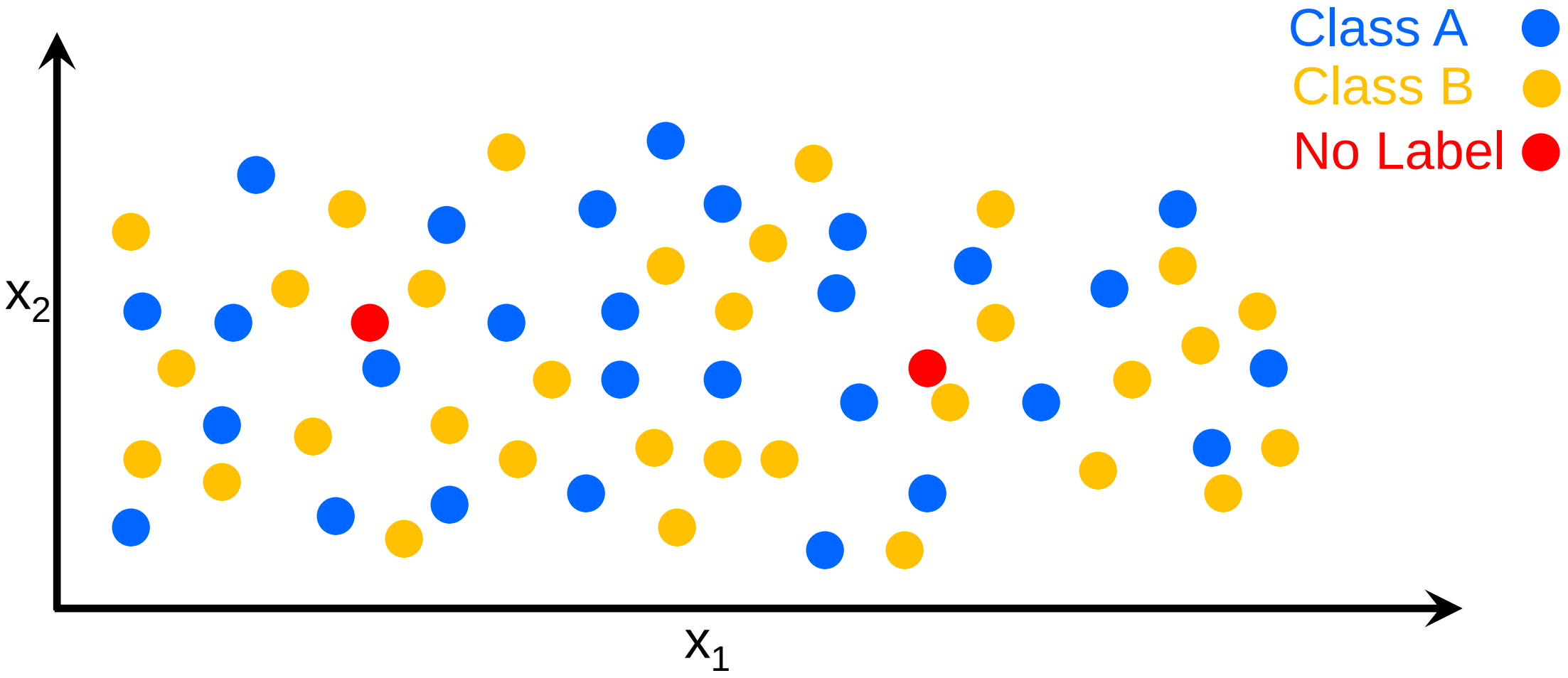


7 – Nearest neighbor

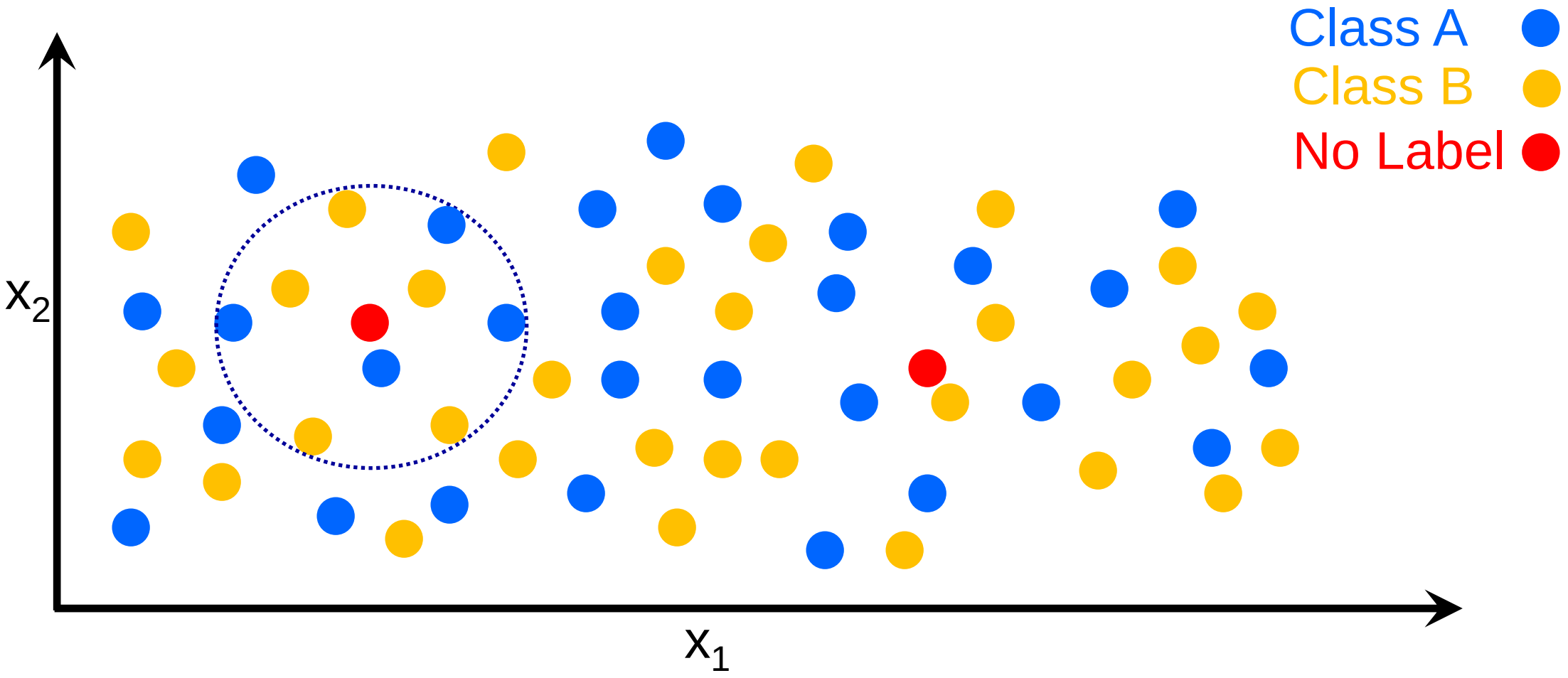


9 – NEAREST NEIGHBOR

9 – Nearest neighbor



9 – Nearest neighbor



CÁC CÂU HỎI

Các câu hỏi

- Câu 01: Hồ Thái Ngọc Nếu số lượng nhãn bằng nhau thì chọn nhãn như thế nào?
- Câu 02: Nguyễn Trần Hoàng Thanh Thuật toán kNN để làm gì?
- Câu 03: Phan Vĩnh Long Chọn k như thế nào tốt nhất?
- Câu 04: Nguyễn Cường Phát Với một bộ dữ liệu cụ thể thì làm sao tìm được k tốt nhất?
- Câu 05: Võ Thị Một, Võ Tuấn Dĩ Làm sao tính được khoảng cách giữa hai điểm Data Point.

DATASET

Dataset

- Tên tập dữ liệu: Social Network Ads.
- Nguồn: https://www.superdatascience.com/pages/machine-learning.
- Tập dữ liệu cho biết các thông tin của khách hàng và họ có mua hàng hay không.

Dataset

- Tập dữ liệu chứa 400 điểm dữ liệu (datapoint), mỗi điểm dữ liệu có 5 thuộc tính gồm:
  + UserID: Mã số định danh của người dùng.
  + Gender: Giới tính của người dùng.
  + Age: Độ tuổi người dùng.
  + Estimated Salary: Mức lương ước đoán của người dùng.
  + Purchased: là một trong hai số 0 và 1. Số 0 cho biết khách hàng không mua hàng và số 1 cho biết khách hàng có mua hàng.

Dataset

– Dưới đây là 5 điểm dữ liệu ngẫu nhiên trong tập dữ liệu.

UserID	Gender	Age	Estimated Salary	Purchased
15624510	Male	19	19,000	0
15810944	Male	35	20,000	1
15668575	Female	26	43,000	0
15603246	Female	27	57,000	0
15804002	Male	19	76,000	1

Dataset

- Bài toán: với 2 thuộc tính:
  - + Độ tuổi (Age)
  - + Mức lương ước đoán (Estimated Salary)

Dự đoán khách hàng sẽ mua hàng hay không?

TIỀN XỬ LÝ DỮ LIỆU

Tiền xử lý dữ liệu

– Ở bài này, ta chỉ quan tâm đến hai thuộc tính tuổi và mức lương ước đoán.

```
1. import pandas as pd
2. import numpy as np
3. dataset = pd.read_csv("Social_Network_Ads.csv")
4. X = dataset.iloc[:, [2, 3]].values
5. Y = dataset.iloc[:, 4].values
```

Tiền xử lý dữ liệu

- Để thuận tiện cho trực quan hóa kết quả sau khi huấn luyện, ta chuẩn hóa dữ liệu về dạng:
  + Kỳ vọng bằng 0.
  + Phương sai bằng 1.
- Lớp `StandardScaler` trong module `sklearn.preprocessing` đã được xây dựng sẵn để chuẩn hóa dữ liệu.

```
7. from sklearn.preprocessing import StandardScaler
8. SC = StandardScaler()
9. X = SC.fit_transform(X)
```

Tiền xử lý dữ liệu

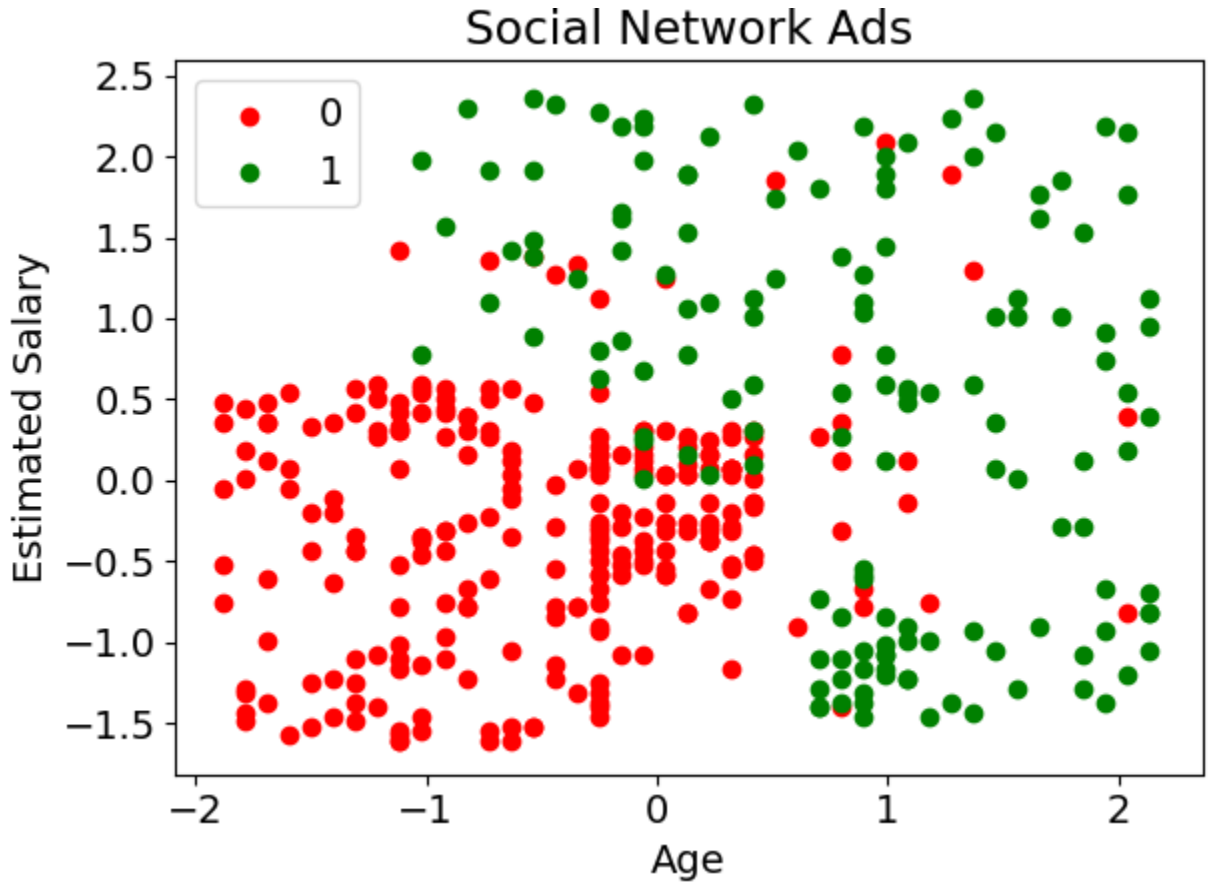
- Chia dữ liệu thành hai tập training set và test set.
- Ta dùng hàm `train_test_split` được cung cấp trong module `sklearn.model_selection`.

```
10.from sklearn.model_selection import train_test_split
11.X_train, X_test, Y_train, Y_test =

train_test_split(X, Y, train_size = 0.8, random_state =
 0)
```

TRỰC QUAN HÓA DỮ LIỆU

Trực quan hóa dữ liệu



Trực quan hóa dữ liệu

– Xây dựng hàm trực quan hóa các điểm dữ liệu.

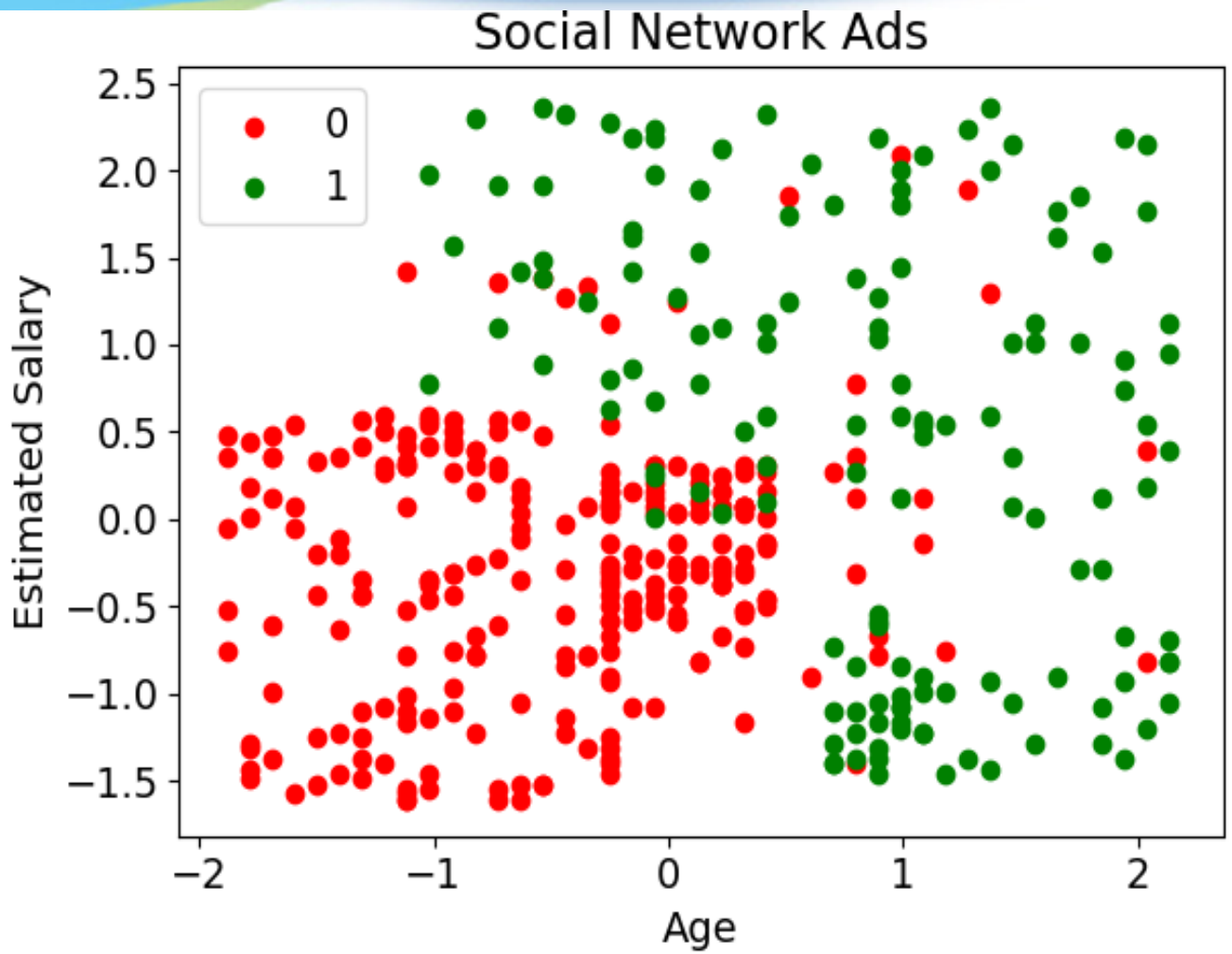
```
11.from matplotlib.colors import ListedColormap
12.import matplotlib.pyplot as plt
13.def VisualizingDataset(X_, Y_):
14.    X1 = X_[:, 0]
15.    X2 = X_[:, 1]
16.    for i, label in enumerate(np.unique(Y_)):
17.        plt.scatter(X1[Y_ == label], X2[Y_ == label]
    ,
```

Trực quan hóa dữ liệu

– Gọi hàm trực quan hóa dữ liệu.

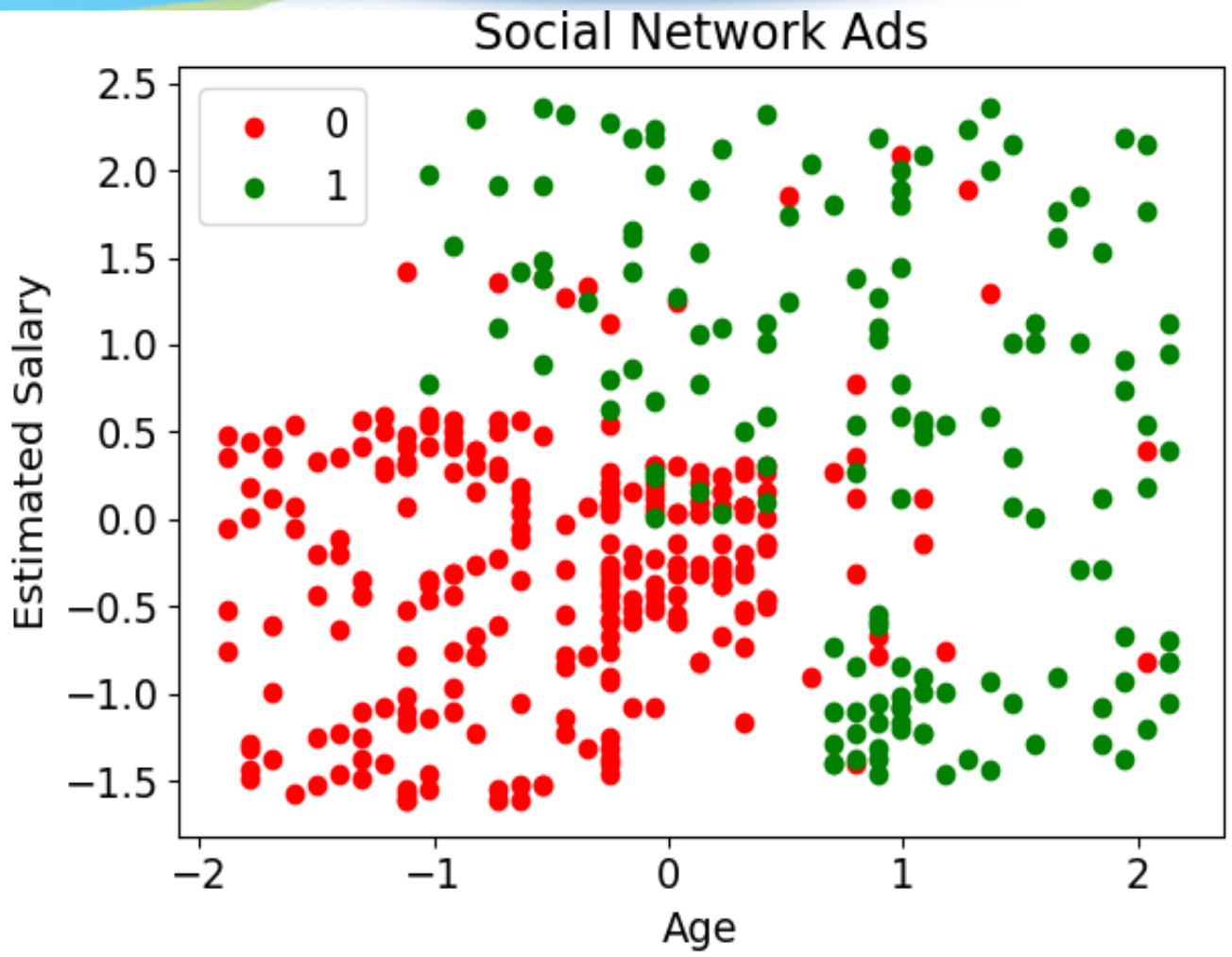
```
18.VisualizingDataset(X, Y)
19.plt.show()
```

Trực quan hóa dữ liệu



- Theo hình vẽ, ta thấy các điểm có sự phân bố thành 2 mảng.
  - + Mảng dưới trái phần lớn có màu đỏ, tức khách hàng không mua hàng.
  - + Mảng bên phải và mảng bên trên phần lớn có màu xanh, tức khách hàng có mua hàng.

Trực quan hóa dữ liệu



- Điều này là phù hợp vì các khách hàng trẻ và có mức lương thấp sẽ thường không mua hàng.
- Ngược lại, khách hàng cao tuổi hoặc có lương cao sẽ thường mua hàng nhiều hơn.

K NEAREST NEIGHBORS

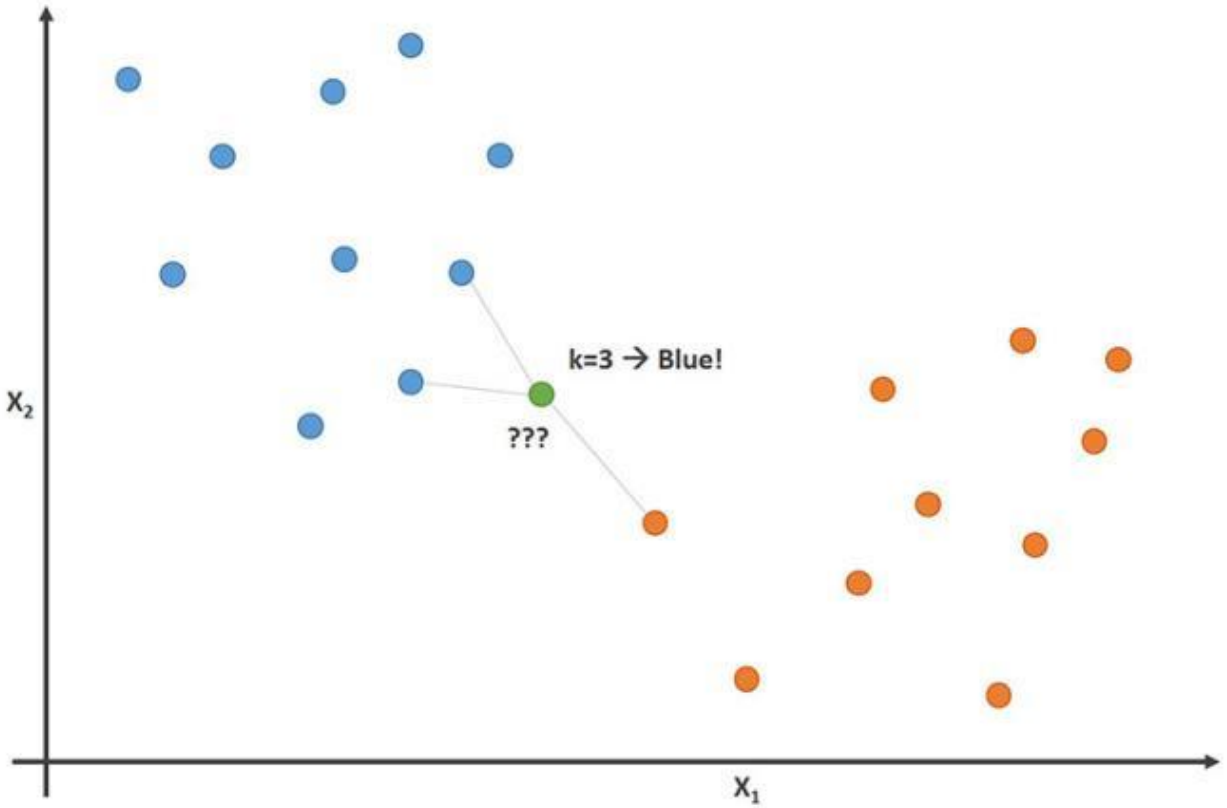
K Nearest Neighbors

- Trong bài toán này, kết quả đầu ra của dữ liệu không còn là một số bất kỳ nữa, mà là một trong hai số 0 hoặc 1.
  + Giá trị 0: không mua hàng.
  + Giá trị 1: mua hàng.
- Đây là bài toán phân loại (*classification*).
- *K Nearest Neighbors* là một kỹ thuật sử dụng cho các bài toán phân loại.

K Nearest Neighbors

- *K Nearest Neighbors* là một trong các thuật toán phân loại đơn giản nhất.
- Thuật toán này sẽ tìm k điểm có khoảng cách gần với điểm cần dự đoán nhất. Nhãn đầu ra của điểm cần dự đoán là nhãn có số lần xuất hiện nhiều nhất trong k điểm được chọn.

K Nearest Neighbors



- Nói cách khác, K Nearest Neighbors dự đoán đầu ra của điểm dữ liệu mới bằng cách dựa trên đầu ra các điểm dữ liệu hàng xóm của nó.

K Nearest Neighbors

- Khoảng cách để xác định hai điểm có phải là hàng xóm hay không có thể là:
  + Khoảng cách Minkowski
  + Khoảng cách Euclid
  + Khoảng cách Manhattan
  + Khoảng cách Chebyshev
  + Độ đo tương tự Cosine.
  + ...

HUẤN LUYỆN MÔ HÌNH

Huấn luyện mô hình

- Ta sử dụng lớp `KNeighborsClassifier` trong module `sklearn.neighbors` để huấn luyện mô hình.
- Số lượng điểm cần tìm – k được đặt là 5.

```
20.from sklearn.neighbors import KNeighborsClassifier
21.classifier = KNeighborsClassifier(n_neighbors= 5)
22.classifier.fit(X_train, Y_train)
```

TRỰC QUAN HÓA KẾT QUẢ MÔ HÌNH

Trực quan hóa kết quả mô hình

- Ta tạo một *confusion matrix*. Đây là một ma trận có kích thước là $p \times p$ với p là số phân lớp trong bài toán đang xét, ở đây là 2.
- Phần tử ở dòng thứ i, cột thứ j của confusion matrix biểu thị số lượng phần tử có loại là i và được phân vào loại j.
- Hàm `confusion_matrix` trong module `sklearn.metrics` sẽ hỗ trợ ta xây dựng confusion matrix.

```
23.from sklearn.metrics import confusion_matrix
24.cm = confusion_matrix(Y_train, classifier.predict(X_
   train))
25.print(cm)
```

Trực quan hóa kết quả mô hình

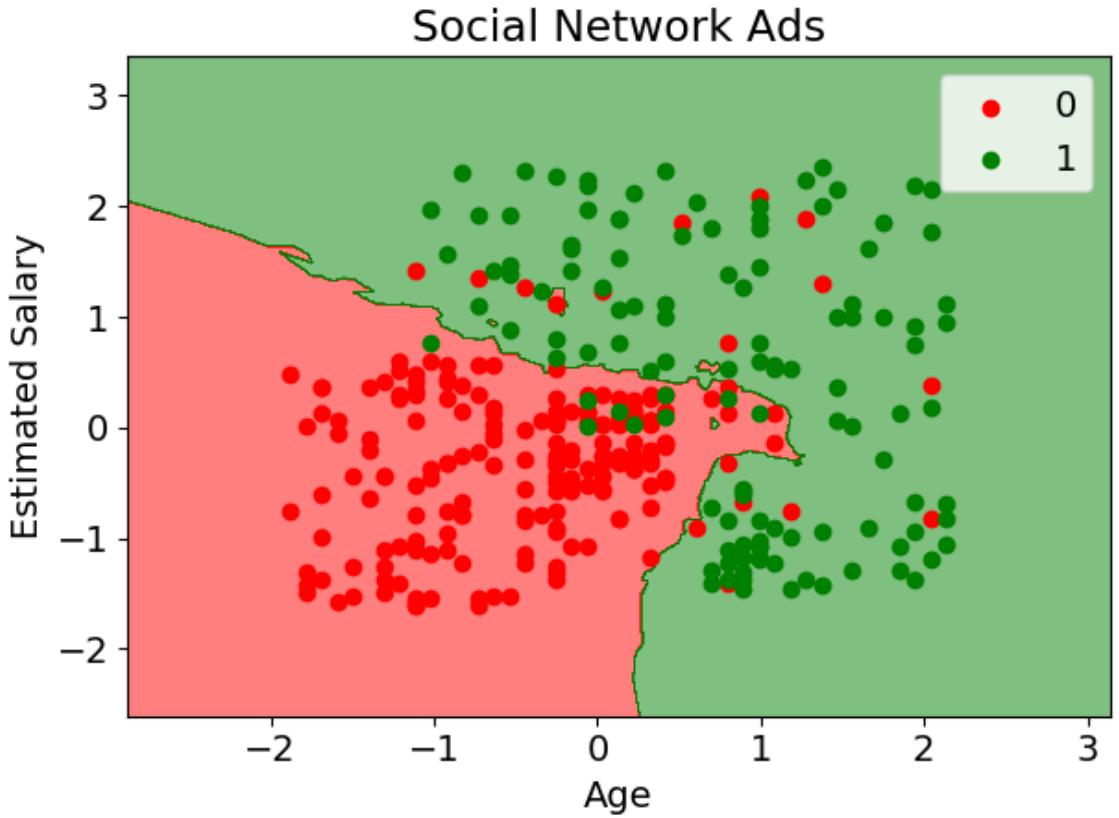
- Confusion Matrix được in ra là:

	0	1
0	182	17
1	9	112

- Theo ma trận trên, số lượng dữ liệu được phân loại đúng là $182 + 112 = 294$ điểm dữ liệu.
- Số lượng dữ liệu phân loại sai là $9 + 17 = 26$ điểm dữ liệu.
- Tỉ lệ điểm dữ liệu phân loại sai là $26/320 = 0.08$.

Trực quan hóa kết quả mô hình

- Ta trực quan hóa kết quả mô hình trên mặt phẳng tọa độ bằng cách vẽ 2 vùng phân chia mà mô hình thu được sau quá trình huấn luyện.



Trực quan hóa kết quả mô hình

– Xây dựng hàm trực quan hóa kết quả bằng cách tạo 2 vùng phân chia mà mô hình đạt được.

```
26.def VisualizingResult(model, X_):
27.     X1 = X_[:, 0]
28.     X2 = X_[:, 1]
29.     X1_range = np.arange(start= X1.min()-1, stop=
   X1.max()+1, step = 0.01)
30.     X2_range = np.arange(start= X2.min()-1, stop=
   X2.max()+1, step = 0.01)
31.     X1_matrix, X2_matrix = np.meshgrid(X1_range, X2_
   range)
```

Trực quan hóa kết quả mô hình

– Xây dựng hàm trực quan hóa kết quả bằng cách tạo 2 vùng phân chia mà mô hình đạt được.

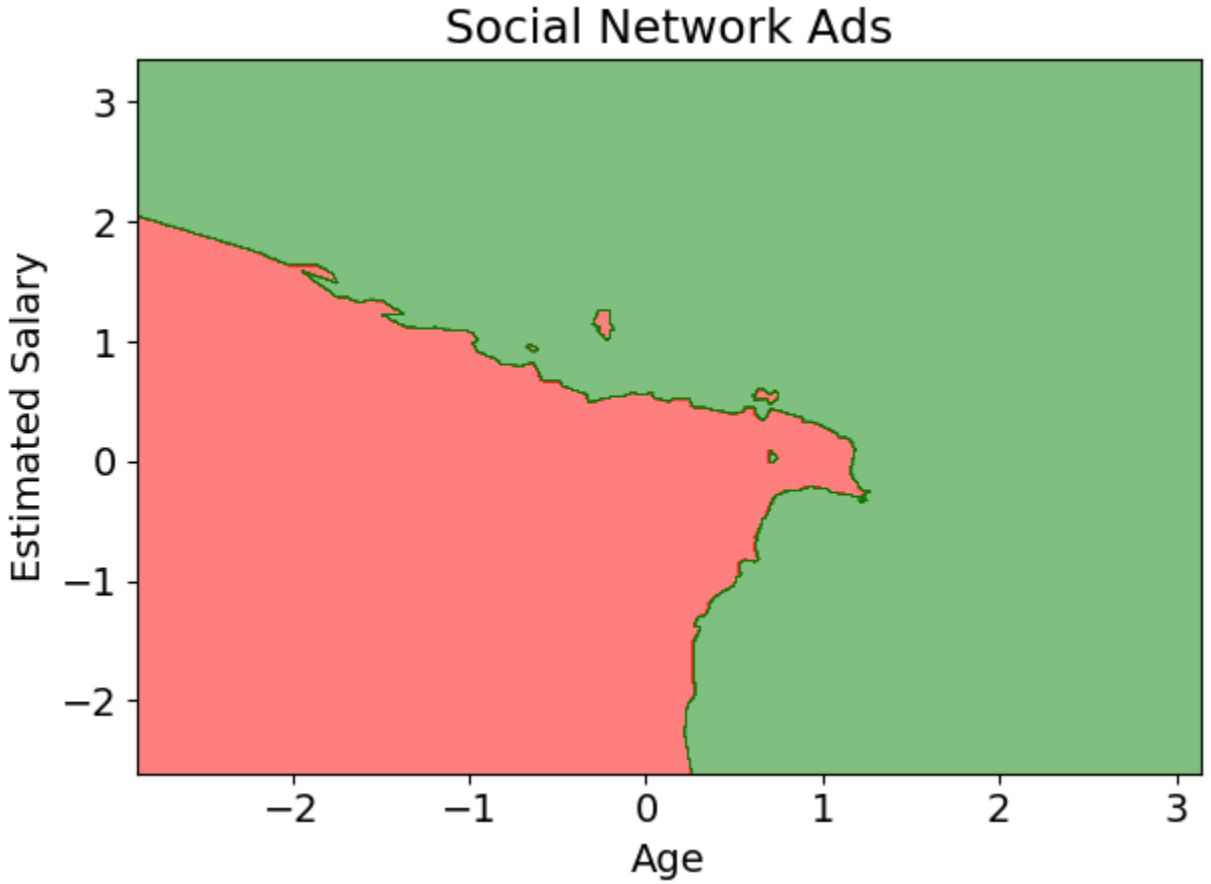
```
26.def VisualizingResult(model, X_):
31.     ...
32.     X_grid= np.array([X1_matrix.ravel(),
   X2_matrix.ravel()]).T
33.     Y_grid=
   model.predict(X_grid).reshape(X1_matrix.shape)
34.     plt.contourf(X1_matrix, X2_matrix, Y_grid, alpha
   = 0.5,    cmap = ListedColormap(("red", "green")))
```

Trực quan hóa kết quả mô hình

– Trực quan hóa kết quả mô hình.

```
35.VisualizingResult(classifier, X_train)
36.plt.show()
```

Trực quan hóa kết quả mô hình

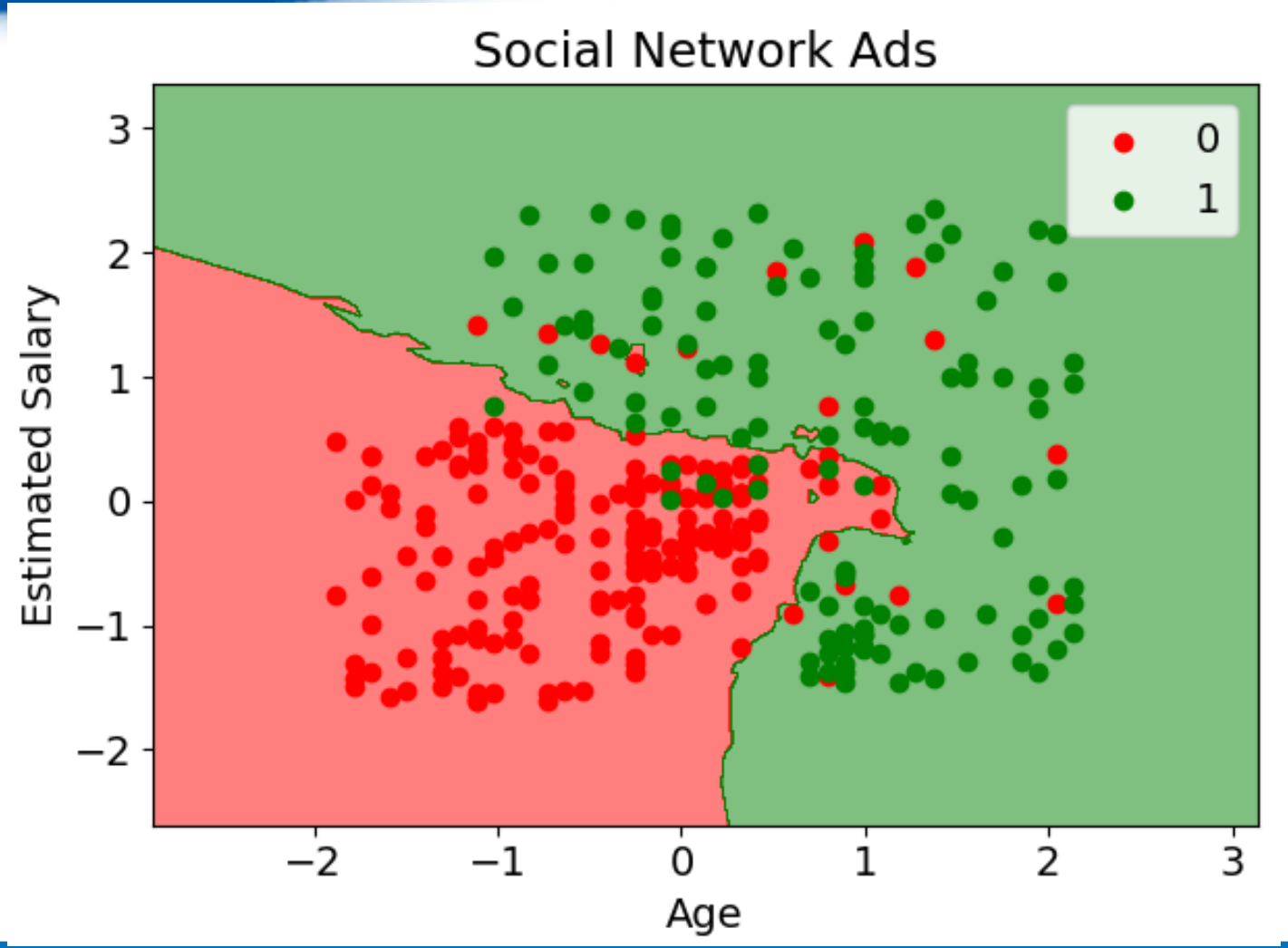


Trực quan hóa kết quả mô hình

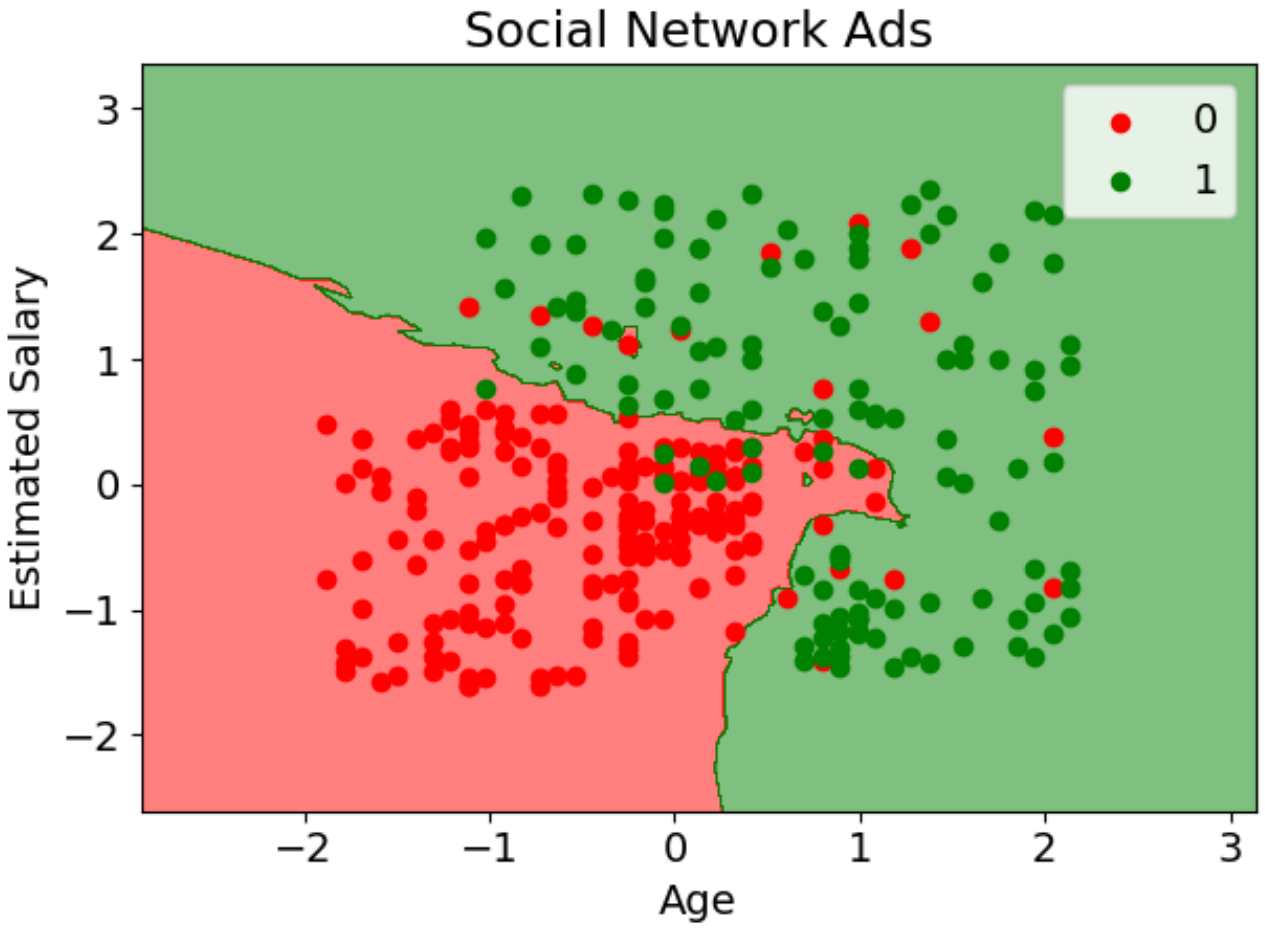
- Hoàn thiện quá trình trực quan bằng cách vẽ thêm các điểm dữ liệu huấn luyện lên mặt phẳng tọa độ.

```
37.VisualizingResult(classifier, X_train)
38.VisualizingDataset(X_train, Y_train)
39.plt.show()
```

Trực quan hóa kết quả mô hình



Trực quan hóa kết quả mô hình



- Nhận xét:
  - + Mô hình phân chia khá tốt.
  - + Đường phân chia có vẻ không có hình dáng ổn định.

KIỂM TRA KẾT QUẢ TRÊN TẬP TEST

Kiểm tra kết quả trên tập test

– Tạo *confusion matrix* trên tập test.

```
40.cm = confusion_matrix(Y_test, classifier.predict(X_t
   est))
41.print(cm)
```

Kiểm tra kết quả trên tập test

- Confusion Matrix được in ra là:

	0	1
0	55	3
1	1	21

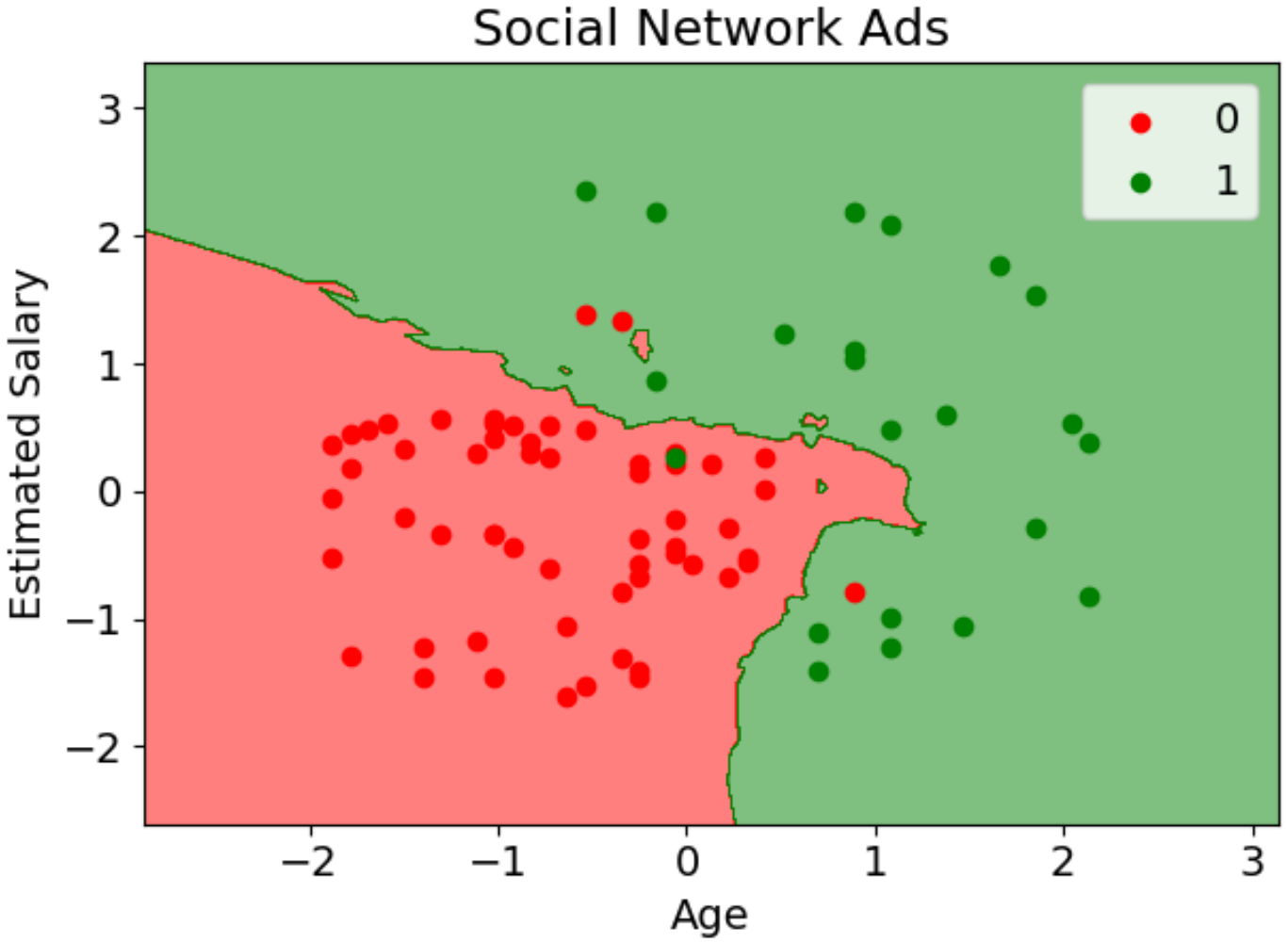
- Theo ma trận trên, số lượng dữ liệu được phân loại đúng là $55 + 21 = 76$ điểm dữ liệu.
- Số lượng dữ liệu phân loại sai là $1 + 3 = 4$ điểm dữ liệu.
- Tỉ lệ điểm dữ liệu phân loại sai là $4/80 \approx 0.05$.

Kiểm tra kết quả trên tập test

– Thực hiện tương tự trực quan hóa kết quả mô hình trên tập traning.

```
42.VisualizingResult(classifier, X_test)
43.VisualizingDataset(X_test, Y_test)
44.plt.show()
```

Kiểm tra kết quả trên tập test



	0	1
0	55	3
1	1	21

Kiểm tra kết quả trên tập test

– Xây dựng hàm so sánh kết quả trên một điểm dữ liệu trong tập test.

```
45.def compare(i_example):
46.     x = X_test[i_example : i_example + 1]
47.     y = Y_test[i_example]
48.     y_pred = classifier.predict(x)
49.     x_inv = SC.inverse_transform(x)
50.     print(x_inv, y, y_pred)
```

Kiểm tra kết quả trên tập test

– Gọi thực hiện hàm so sánh trên 5 điểm dữ liệu, có chỉ mục từ thứ 7 đến 11 trong tập kiểm thử.

```
51.for i in range(7, 12):
52.     compare(i)
```

Kiểm tra kết quả trên tập test

Age	Estimated Salary	Purchased	Predicted Purchased
36	144,000	1	1
18	68,000	0	0
47	43,000	0	1
30	49,000	0	0
28	53,000	0	0

Chúc các bạn học tốt

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN TP.HCM

Nhóm UIT-Together

Nguyễn Tấn Trần Minh Khang