

IMAGE CLASSIFICATION PIPELINE

- ThS. Đoàn Chánh Thống
- ThS. Nguyễn Cường Phát
- ThS. Nguyễn Hữu Lợi
- ThS. Trương Quốc Dũng
- ThS. Nguyễn Thành Hiệp
- ThS. Võ Duy Nguyên
- ThS. Nguyễn Văn Toàn
- ThS. Lê Ngô Thực Vi
- TS. Nguyễn Duy Khánh
- TS. Nguyễn Tấn Trần Minh Khang

IMAGE CLASSIFICATION PROBLEM

Image Classification

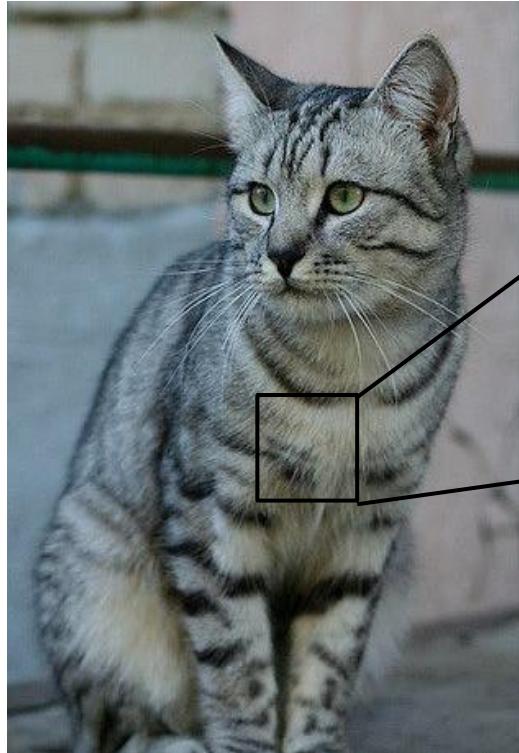
- Image Classification: A core task in Computer Vision.
- Phân loại ảnh: Tác vụ (bài toán) cốt lõi trong Thị giác Máy tính.



cat

(assume given set of discrete labels)
{dog, cat, truck, plane, ...}

The Problem: Semantic Gap



```
[[105 112 108 111 104 99 106 99 96 103 112 119 104 97 93 87]
 [ 91 98 102 106 104 79 98 103 99 105 123 136 110 105 94 85]
 [ 76 85 90 105 128 105 87 96 95 99 115 112 106 103 99 85]
 [ 99 81 81 93 120 131 127 100 95 98 102 99 96 93 101 94]
 [106 91 61 64 69 91 88 85 101 107 109 98 75 84 96 95]
 [114 108 85 55 55 69 64 54 64 87 112 129 98 74 84 91]
 [133 137 147 103 65 81 80 65 52 54 74 84 102 93 85 82]
 [128 137 144 140 109 95 86 70 62 65 63 63 60 73 86 101]
 [125 133 148 137 119 121 117 94 65 79 80 65 54 64 72 98]
 [127 125 131 147 133 127 126 131 111 96 89 75 61 64 72 84]
 [115 114 109 123 150 148 131 118 113 109 100 92 74 65 72 78]
 [ 89 93 90 97 108 147 131 118 113 114 113 109 106 95 77 80]
 [ 63 77 86 81 77 79 102 123 117 115 117 125 125 130 115 87]
 [ 62 65 82 89 78 71 80 101 124 126 119 101 107 114 131 119]
 [ 63 65 75 88 89 71 62 81 120 138 135 105 81 98 110 118]
 [ 87 65 71 87 106 95 69 45 76 130 126 107 92 94 105 112]
 [118 97 82 86 117 123 116 66 41 51 95 93 89 95 102 107]
 [164 146 112 80 82 120 124 104 76 48 45 66 88 101 102 109]
 [157 170 157 120 93 86 114 132 112 97 69 55 70 82 99 94]
 [130 128 134 161 139 100 109 118 121 134 114 87 65 53 69 86]
 [128 112 96 117 150 144 120 115 104 107 102 93 87 81 72 79]
 [123 107 96 86 83 112 153 149 122 109 104 75 80 107 112 99]
 [122 121 102 80 82 86 94 117 145 148 153 102 58 78 92 107]
 [122 164 148 103 71 56 78 83 93 103 119 139 102 61 69 84]]
```

An image is just a big grid of numbers between [0, 255]:

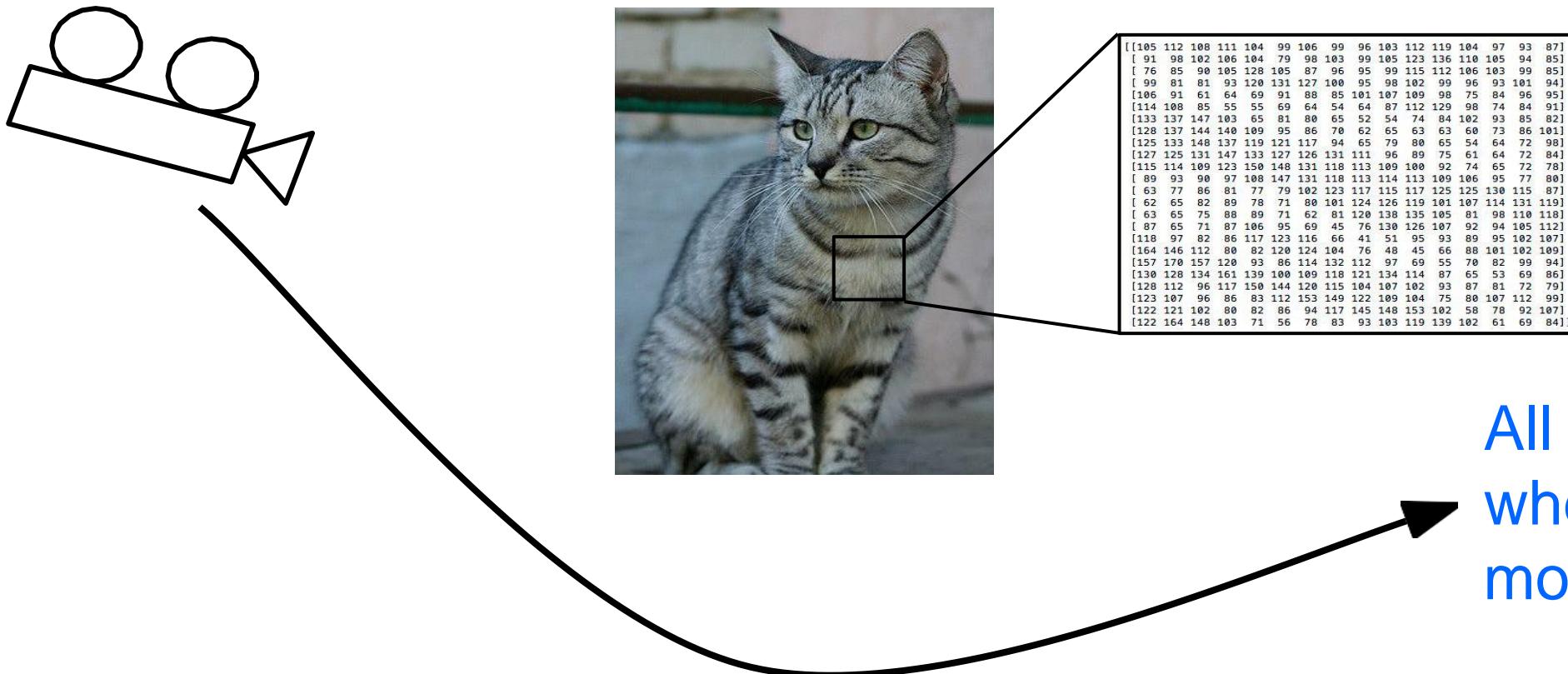
e.g. $800 \times 600 \times 3 = 1,440,000$
(3 channels RGB)

CHALLENGES

Challenges

1. Viewpoint variation – Góc nhìn khác nhau.
2. Illumination – Độ sáng.
3. Deformation – Độ biến dạng.
4. Occlusion – Sự che khuất – Sự che lấp.
5. Background Clutter – Nhiễu nền.
6. Intraclass variation – Sự đa dạng cục bộ trong một lớp.

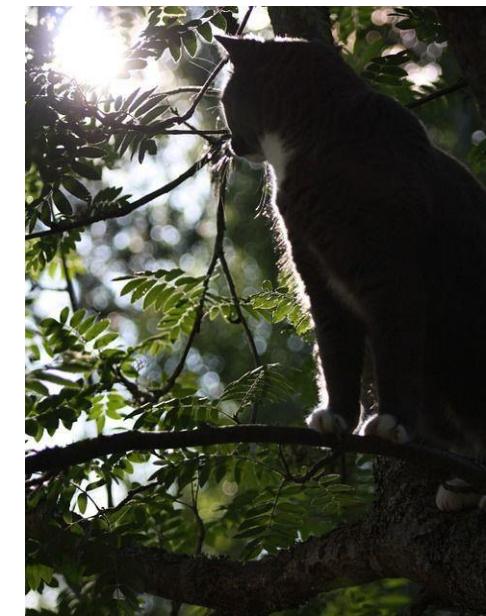
Challenges: Viewpoint variation



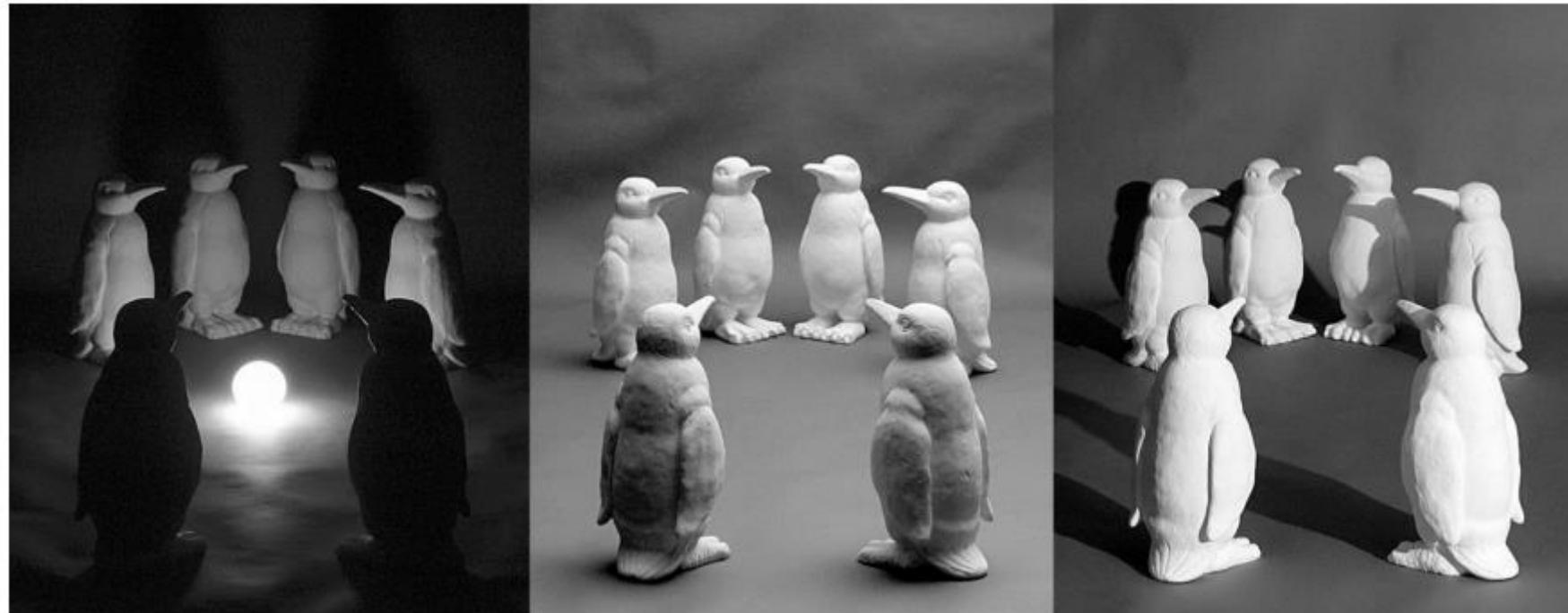
Challenges: Viewpoint variation



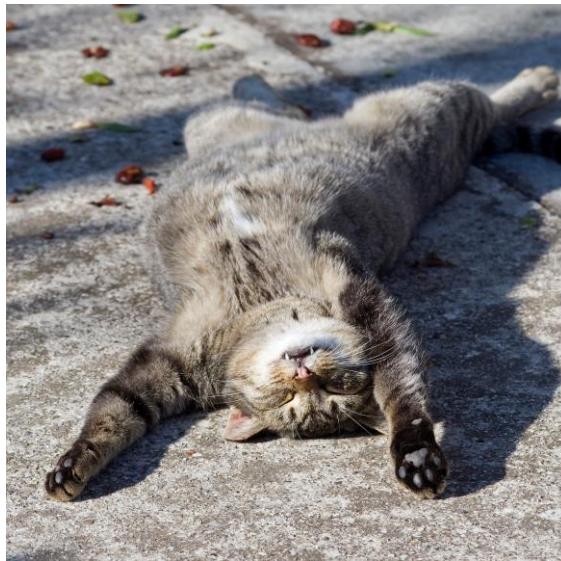
Challenges: Illumination



Challenges: Illumination



Challenges: Deformation



Challenges: Occlusion



Challenges: Background Clutter



Challenges: Intraclass variation



CÂU HỎI ÔN TẬP

Câu hỏi

— Câu hỏi 1: Có bao nhiêu thách thức?

Câu hỏi

- Câu hỏi 1: Có bao nhiêu thách thức?
- Trả lời: Có sáu thách thức.

Câu hỏi

— Câu hỏi 2: Thách thức thứ nhất là gì?

Câu hỏi

- Câu hỏi 2: Thách thức thứ nhất là gì?
- Trả lời: Viewpoint variation – Góc nhìn khác nhau.

Câu hỏi

— Câu hỏi 3: Thách thức thứ hai là gì?

Câu hỏi

- Câu hỏi 3: Thách thức thứ hai là gì?
- Trả lời: Illumination – Độ sáng.

Câu hỏi

— Câu hỏi 4: Thách thức thứ ba là gì?

Câu hỏi

- Câu hỏi 4: Thách thức thứ ba là gì?
- Trả lời: Deformation – Độ biến dạng.

Câu hỏi

— Câu hỏi 5: Thách thức thứ tư là gì?

Câu hỏi

- Câu hỏi 5: Thách thức thứ tư là gì?
- Trả lời: Occlusion – Sự che khuất – Sự che lấp.

Câu hỏi

— Câu hỏi 6: Thách thức thứ năm là gì?

Câu hỏi

- Câu hỏi 6: Thách thức thứ năm là gì?
- Trả lời: Background Clutter – Nhiễu nền.

Câu hỏi

— Câu hỏi 7: Thách thức thứ sáu là gì?

Câu hỏi

- Câu hỏi 7: Thách thức thứ sáu là gì?
- Trả lời: Intraclass variation – Sự đa dạng cục bộ trong một lớp.

Câu hỏi

– Câu hỏi 8: Bài toán Image Classification dịch ra tiếng Việt là gì?

Câu hỏi

- Câu hỏi 8: Bài toán Image Classification dịch ra tiếng Việt là gì?
- Trả lời: Bài toán Image Classification dịch ra tiếng Việt là Bài toán phân lớp ảnh.

Câu hỏi

– Câu hỏi 9: Đầu vào của bài toán phân lớp ảnh là gì?

Câu hỏi

- Câu hỏi 9: Đầu vào của bài toán phân lớp ảnh là gì?
- Trả lời: Đầu vào của bài toán phân lớp ảnh là một hoặc nhiều ảnh cần phân lớp.

Câu hỏi

– Câu hỏi 10: Đầu ra của bài toán phân lớp ảnh là gì?

Câu hỏi

- Câu hỏi 10: Đầu ra của bài toán phân lớp ảnh là gì?
- Trả lời: Đầu ra của bài toán phân lớp ảnh là một hoặc nhiều nhãn tương ứng với các ảnh đầu vào.

Câu hỏi

— Câu hỏi 11: Ảnh màu trong máy tính thông thường có mấy kênh?

Câu hỏi

- Câu hỏi 11: Ảnh màu trong máy tính thông thường có mấy kênh?
- Trả lời: Ảnh màu trong máy tính thông thường có ba kênh là red, green, blue.

Câu hỏi

- Câu hỏi 12: Mỗi pixel trong ảnh màu được biểu diễn thông qua mấy giá trị?

Câu hỏi

- Câu hỏi 12: Mỗi pixel trong ảnh màu được biểu diễn thông qua mấy giá trị?
- Trả lời: Mỗi pixel trong ảnh màu được biểu diễn thông qua ba giá trị tương ứng với ba kênh màu là red, green, blue.

Câu hỏi

— Câu hỏi 13: Thông thường bên trong máy tính giá trị 0 biểu diễn màu nào?

Câu hỏi

- Câu hỏi 13: Thông thường bên trong máy tính giá trị 0 biểu diễn màu nào?
- Trả lời: Thông thường bên trong máy tính giá trị 0 biểu diễn màu đen.

Câu hỏi

- Câu hỏi 14: Thông thường bên trong máy tính giá trị 255 biểu diễn màu nào?

Câu hỏi

- Câu hỏi 14: Thông thường bên trong máy tính giá trị 255 biểu diễn màu nào?
- Trả lời: Thông thường bên trong máy tính giá trị 255 biểu diễn màu trắng.

Câu hỏi

- Câu hỏi 15: Trong ma trận biểu diễn một kênh màu (red hoặc green hoặc blue) của một ảnh màu các giá trị nằm trong phạm vi nào?

Câu hỏi

- Câu hỏi 15: Trong ma trận biểu diễn một kênh màu (red hoặc green hoặc blue) của một ảnh màu các giá trị nằm trong phạm vi nào?
- Trả lời: Trong ma trận biểu diễn một kênh màu (red hoặc green hoặc blue) của một ảnh màu các giá trị nằm trong phạm vi từ 0 đến 255.

AN IMAGE CLASSIFIER

An image classifier

```
def classify_image( image ):  
    # Some magic here?  
    return class_label
```

An image classifier

```
def classify_image(image):  
    # Some magic here?  
    return class_label
```

- Unlike e.g. sorting a list of numbers,
- Không giống như thuật toán sắp xếp,

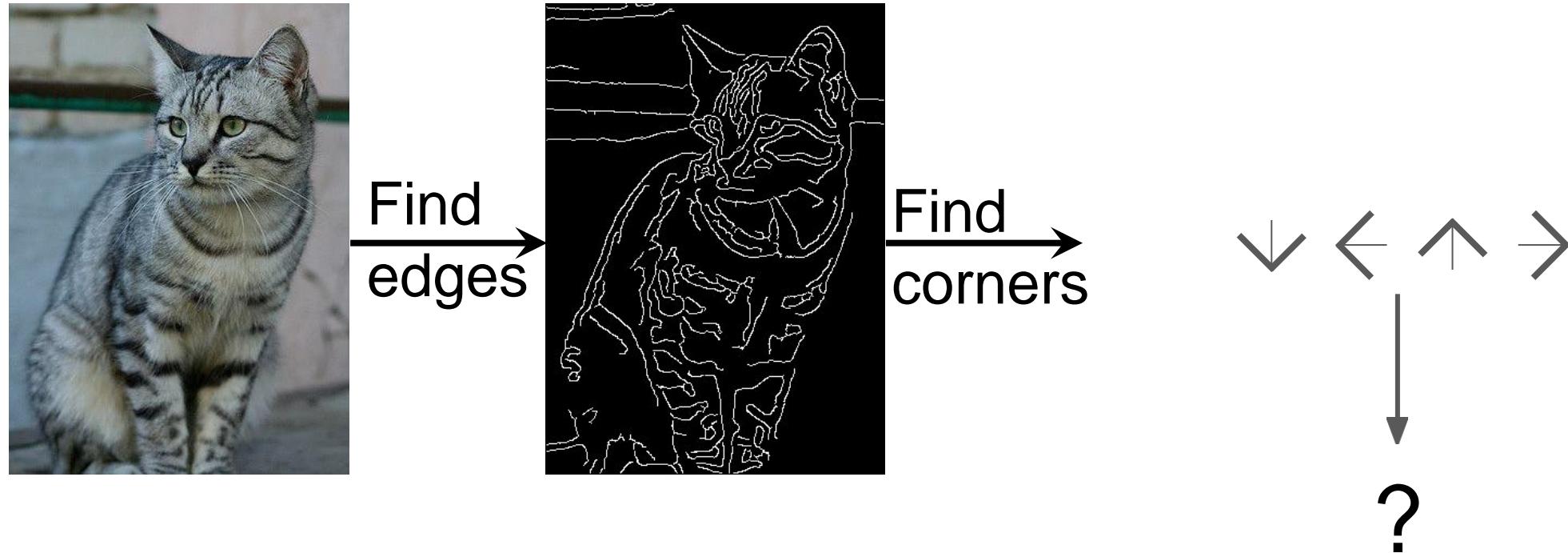
An image classifier

```
def classify_image(image):  
    # Some magic here?  
    return class_label
```

- no obvious way to hard – code the algorithm for recognizing a cat, or other classes.
- không thuật toán rõ ràng để nhận dạng một con mèo hoặc các lớp nhãn khác.

ATTEMPTS HAVE BEEN MADE

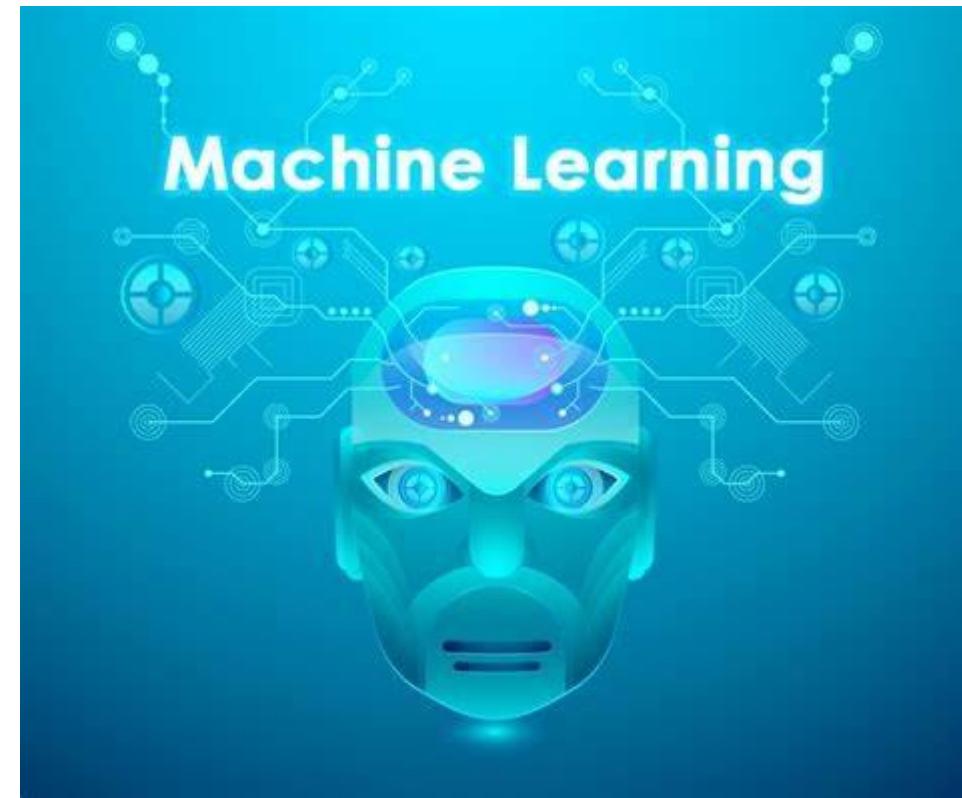
Attempts have been made



MACHINE LEARNING

Data – Driven Approach

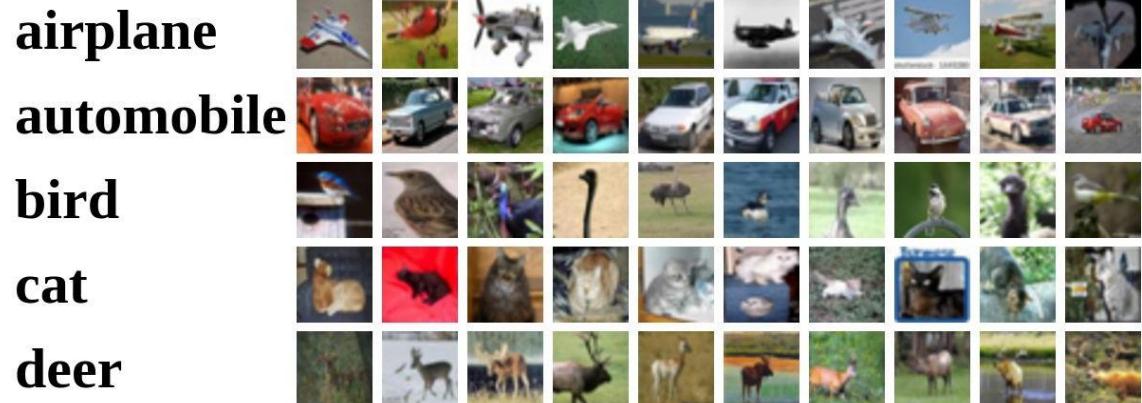
1. Collect a dataset of images and labels
2. Use Machine Learning to train a classifier
3. Evaluate the classifier on new images



Data – Driven Approach

1. Collect a dataset of images and labels
2. Use Machine Learning to train a classifier
3. Evaluate the classifier on new images

Example training set



Data – Driven Approach

1. Collect a dataset of images and labels
2. Use Machine Learning to train a classifier
3. Evaluate the classifier on new images

```
def train(images, labels):  
    # Machine learning!  
    return model
```

```
def predict(model, test_images):  
    # Use model to predict labels  
    return test_labels
```

Data – Driven Approach

1. Collect a dataset of images and labels
2. Use Machine Learning to train a classifier
3. Evaluate the classifier on new images

```
def train(images, labels):  
    # Machine learning!  
    return model
```

```
def predict(model, test_images):  
    # Use model to predict labels  
    return test_labels
```

CÂU HỎI ÔN TẬP

Câu hỏi

— Câu hỏi 01: Data – Driven Approach có mấy bước?

Câu hỏi

- Câu hỏi 01: Data – Driven Approach có mấy bước?
- Trả lời: Có ba bước.

Câu hỏi

- Câu hỏi 02: Bước thứ nhất của Data – Driven Approach là gì?

Câu hỏi

- Câu hỏi 02: Bước thứ nhất của Data – Driven Approach là gì?
- Trả lời: Thu thập dữ liệu và gán nhãn.

Câu hỏi

— Câu hỏi 03: Bước thứ hai của Data – Driven Approach là gì?

Câu hỏi

- Câu hỏi 03: Bước thứ hai của Data – Driven Approach là gì?
- Trả lời: Sử dụng các thuật toán Machine Learning để xây dựng model.

Câu hỏi

— Câu hỏi 04: Bước thứ ba của Data – Driven Approach là gì?

Câu hỏi

- Câu hỏi 04: Bước thứ ba của Data – Driven Approach là gì?
- Trả lời: Đánh giá Model bằng tập dữ liệu kiểm thử (DataTest).

Câu hỏi

- Câu hỏi 05: Có một thuật toán rõ ràng để phân lớp một ảnh hay không?

Câu hỏi

- Câu hỏi 05: Có một thuật toán rõ ràng để phân lớp một ảnh hay không?
- Trả lời: Không có một thuật toán rõ ràng để phân lớp một ảnh.

Câu hỏi

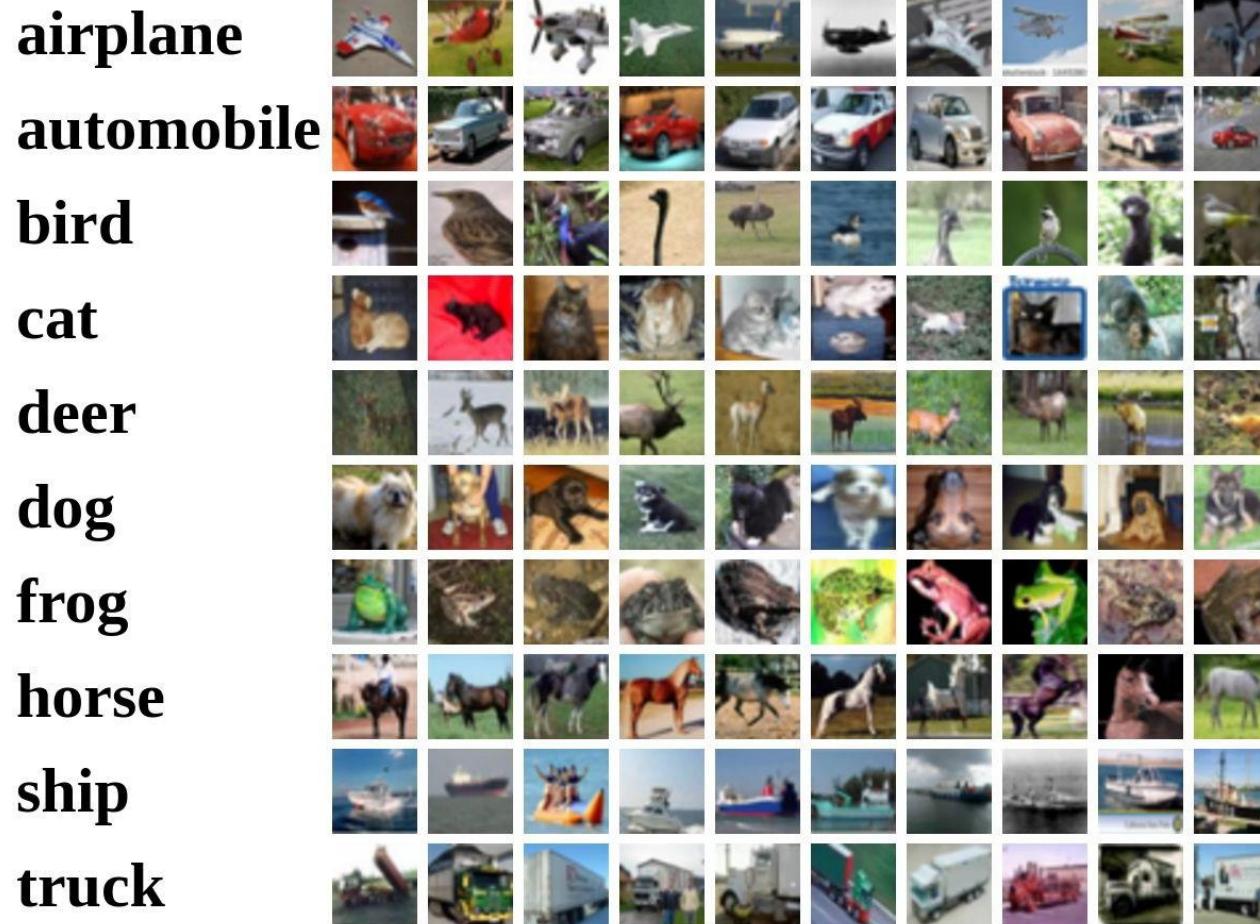
- Câu hỏi 06: Các nỗ lực đã được thực hiện để giải quyết bài toán phân lớp một ảnh là gì?

Câu hỏi

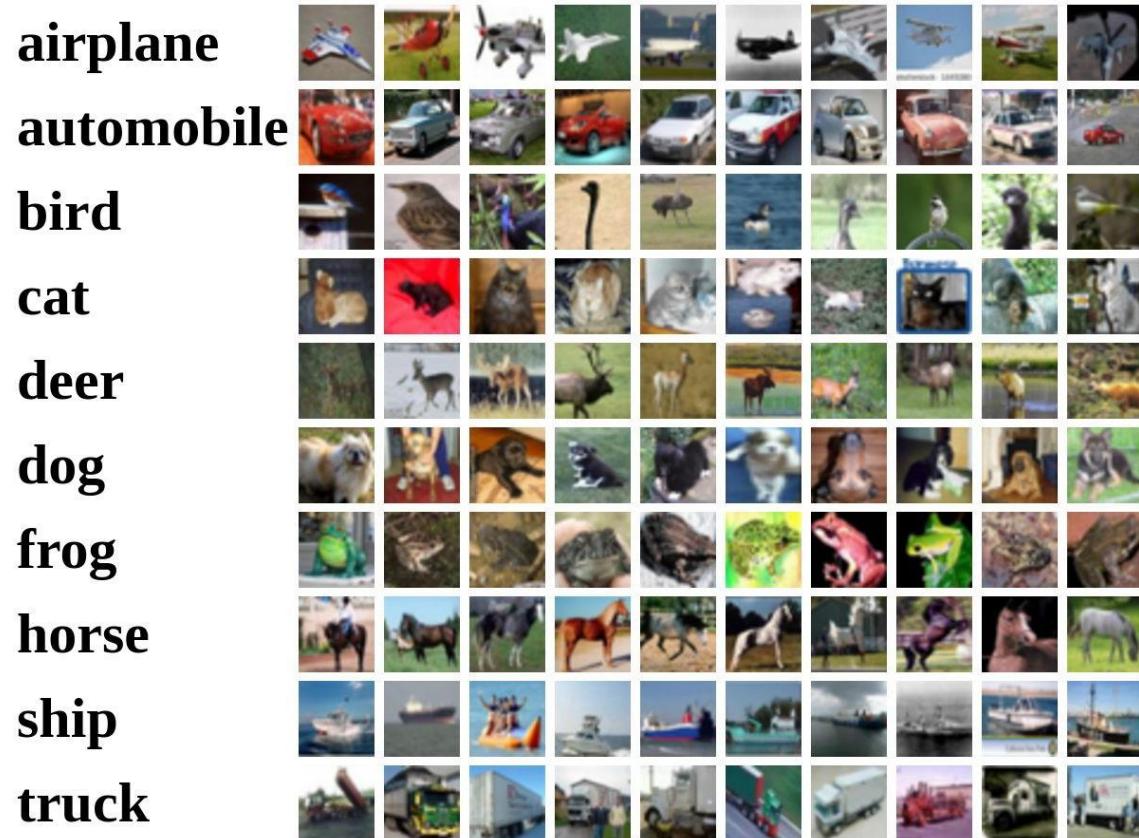
- Câu hỏi 06: Các nỗ lực đã được thực hiện để giải quyết bài toán phân lớp một ảnh là gì?
- Trả lời: Các nỗ lực đã được thực hiện để giải quyết bài toán phân lớp một ảnh là phát hiện các cạnh, các góc, các contour, các cấu trúc hoặc dữ liệu thống kê,....

CIFAR10

Dataset: CIFAR10



Dataset: CIFAR10



— Lịch sử:

- + Công bố 2009.
- + Bộ dữ liệu Cifar10 được phát triển bởi nhóm nghiên cứu tại Đại học Toronto.
- + Nhóm tác giả: Alex Krizhevsky, Vinod Nair và Geoffrey Hinton.

Dataset: CIFAR10

airplane



automobile



bird



cat



deer



dog



frog



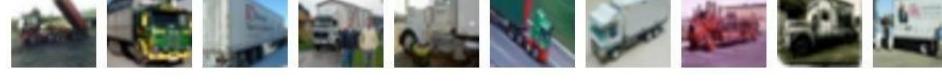
horse



ship



truck



— 10 classes.

*airplane
automobile
bird
cat
deer
dog
frog
horse
ship
truck*

Dataset: CIFAR10

airplane



automobile



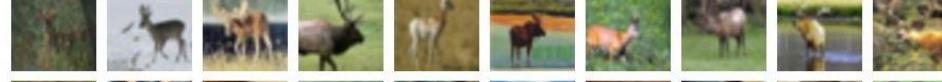
bird



cat



deer



dog



frog



horse



ship



truck

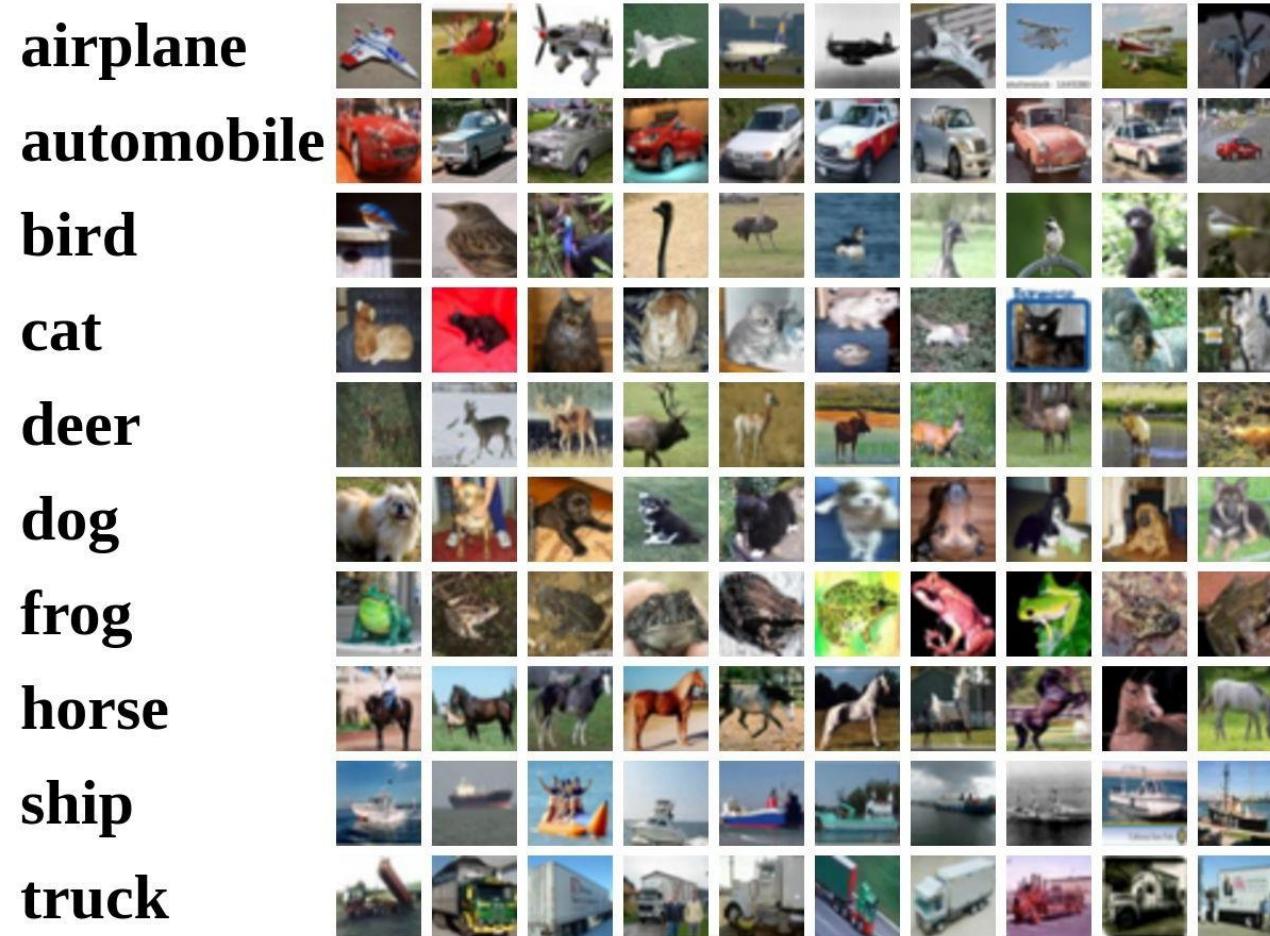


– 10 classes.

*airplane
automobile
bird
cat
deer
dog
frog
horse
ship
truck*

$$= \begin{pmatrix} 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \end{pmatrix}$$

Dataset: CIFAR10



– Datatrain of CIFAR10:

$$\mathcal{D}_{train} = \{(x^{(i)}, y^{(i)})\}_{i=0}^{N-1}$$

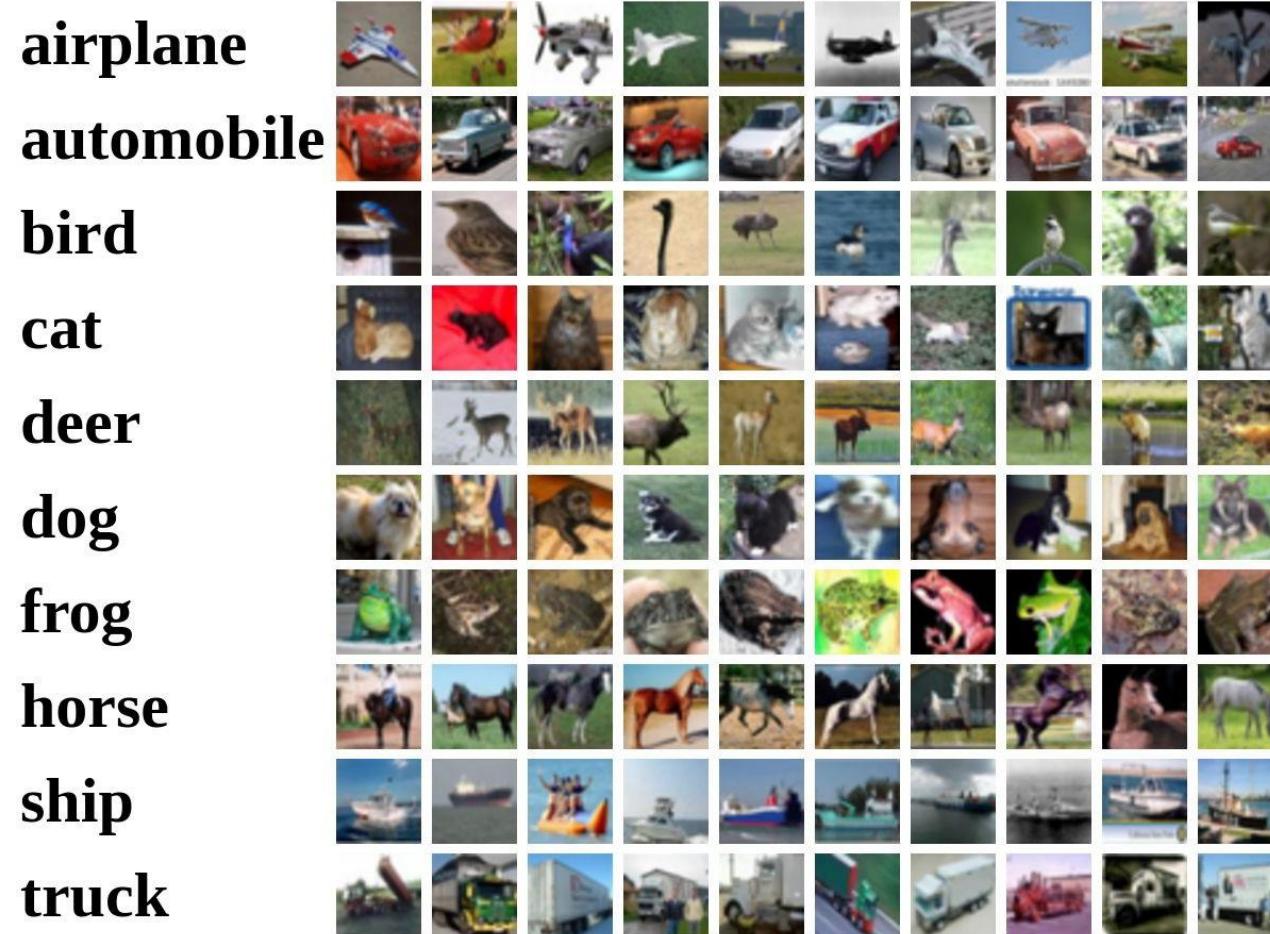
– Where:

+ N : 50.000.

+ $x^{(i)}$ ảnh train thứ i có kích thước $32 \times 32 \times 3$.

+ $y^{(i)}$ nhãn ảnh thứ $i \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$.

Dataset: CIFAR10



– Datatest of CIFAR10:

$$\mathcal{D}_{test} = \{(x^{(i)}, y^{(i)})\}_{i=0}^{M-1}$$

– Where:

+ M : 10.000.

+ $x^{(i)}$ ảnh test thứ i có kích thước $32 \times 32 \times 3$.

+ $y^{(i)}$ nhãn ảnh thứ $i \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$.

Data – Driven Approach

1. Collect a dataset of images and labels
2. Use Machine Learning to train a classifier
3. Evaluate the classifier on new images

Example training set

airplane



automobile



bird



cat



deer



CÂU HỎI ÔN TẬP

Câu hỏi

— Câu hỏi: Bộ dữ liệu CIFAR 10 được công bố năm nào?

Câu hỏi

- Câu hỏi: Bộ dữ liệu CIFAR 10 được công bố năm nào?
- Trả lời: Bộ dữ liệu CIFAR 10 được công bố năm 2009.

Câu hỏi

- Câu hỏi: Bộ dữ liệu CIFAR 10 được công bố bởi trường đại học nào?

Câu hỏi

- Câu hỏi: Bộ dữ liệu CIFAR 10 được công bố bởi trường đại học nào?
- Trả lời: Bộ dữ liệu CIFAR 10 được công bố bởi trường đại học Toronto.

Câu hỏi

— Câu hỏi: Bộ dữ liệu CIFAR 10 có bao nhiêu lớp nhãn?

Câu hỏi

- Câu hỏi: Bộ dữ liệu CIFAR 10 có bao nhiêu lớp nhãn?
- Trả lời: Bộ dữ liệu CIFAR 10 có 10 lớp nhãn.

Câu hỏi

- Câu hỏi: Bộ dữ liệu CIFAR 10 các lớp nhãn được đánh chỉ số từ đâu đến đâu?

Câu hỏi

- Câu hỏi: Bộ dữ liệu CIFAR 10 các lớp nhãn được đánh chỉ số từ đâu đến đâu?
- Trả lời: Bộ dữ liệu CIFAR 10 các lớp nhãn được đánh chỉ số từ 0 đến 9.

Câu hỏi

— Câu hỏi: Liệt kê tên các lớp nhãn của bộ dữ liệu CIFAR 10?

Câu hỏi

- Câu hỏi: Liệt kê tên các lớp nhãn của bộ dữ liệu CIFAR 10?
- Trả lời: Tên các lớp nhãn của bộ dữ liệu CIFAR 10 là:

(*airplane
automobile
bird
cat
deer
dog
frog
horse
ship
truck*)

Câu hỏi

- Bộ dữ liệu Datatrain tên tiếng Việt là gì?
 - Datatrain of CIFAR10:
$$\mathcal{D}_{train} = \{(x^{(i)}, y^{(i)})\}_{i=0}^{N-1}$$
 - Where:
 - + N : 50.000.
 - + $x^{(i)}$ ảnh train thứ i có kích thước $32 \times 32 \times 3$.
 - + $y^{(i)}$ nhãn ảnh thứ $i \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$.

Câu hỏi

- Bộ dữ liệu Datatrain tên tiếng Việt là gì?
- Bộ dữ liệu Datatrain tên tiếng Việt là bộ dữ liệu huấn luyện.
 - Datatrain of CIFAR10:
$$\mathcal{D}_{train} = \{(x^{(i)}, y^{(i)})\}_{i=0}^{N-1}$$
 - Where:
 - + N : 50.000.
 - + $x^{(i)}$ ảnh train thứ i có kích thước $32 \times 32 \times 3$.
 - + $y^{(i)}$ nhãn ảnh thứ $i \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$.

Câu hỏi

- Bộ dữ liệu Datatrain của CIFAR 10 có bao nhiêu điểm dữ liệu?
 - Datatrain of CIFAR10:
$$\mathcal{D}_{train} = \{(x^{(i)}, y^{(i)})\}_{i=0}^{N-1}$$
 - Where:
 - + N : 50.000.
 - + $x^{(i)}$ ảnh train thứ i có kích thước $32 \times 32 \times 3$.
 - + $y^{(i)}$ nhãn ảnh thứ $i \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$.

Câu hỏi

- Bộ dữ liệu Datatrain của CIFAR 10 có bao nhiêu điểm dữ liệu?
- Bộ dữ liệu Datatrain của CIFAR 10 có 50.000 điểm dữ liệu.

- Datatrain of CIFAR10:
$$\mathcal{D}_{train} = \{(x^{(i)}, y^{(i)})\}_{i=0}^{N-1}$$
- Where:
 - + N : 50.000.
 - + $x^{(i)}$ ảnh train thứ i có kích thước $32 \times 32 \times 3$.
 - + $y^{(i)}$ nhãn ảnh thứ $i \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$.

Câu hỏi

— Bộ dữ liệu Datatrain của CIFAR 10 có kích thước là bao nhiêu?

— Datatrain of CIFAR10:

$$\mathcal{D}_{train} = \{(x^{(i)}, y^{(i)})\}_{i=0}^{N-1}$$

— Where:

+ N : 50.000.

+ $x^{(i)}$ ảnh train thứ i có kích thước $32 \times 32 \times 3$.

+ $y^{(i)}$ nhãn ảnh thứ $i \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$.

Câu hỏi

- Bộ dữ liệu Datatrain của CIFAR 10 có kích thước là bao nhiêu?
- Bộ dữ liệu Datatrain của CIFAR 10 có kích thước 50.000.

- Datatrain of CIFAR10:
$$\mathcal{D}_{train} = \{(x^{(i)}, y^{(i)})\}_{i=0}^{N-1}$$
- Where:
 - + N : 50.000.
 - + $x^{(i)}$ ảnh train thứ i có kích thước $32 \times 32 \times 3$.
 - + $y^{(i)}$ nhãn ảnh thứ $i \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$.

Câu hỏi

- Chỉ số các điểm dữ liệu trong bộ dữ liệu Datatrain của CIFAR 10 được đánh số trong phạm vi nào?
 - Datatrain of CIFAR10:
$$\mathcal{D}_{train} = \{(x^{(i)}, y^{(i)})\}_{i=0}^{N-1}$$
 - Where:
 - + N : 50.000.
 - + $x^{(i)}$ ảnh train thứ i có kích thước $32 \times 32 \times 3$.
 - + $y^{(i)}$ nhãn ảnh thứ $i \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$.

Câu hỏi

- Chỉ số các điểm dữ liệu trong bộ dữ liệu Datatrain của CIFAR 10 được đánh số trong phạm vi nào?
 - Chỉ số các điểm dữ liệu trong bộ dữ liệu Datatrain của CIFAR 10 được đánh số trong phạm vi $0 \rightarrow 49.999$.
- Datatrain of CIFAR10:
- $$\mathcal{D}_{train} = \{(x^{(i)}, y^{(i)})\}_{i=0}^{N-1}$$
- Where:
- + N : 50.000.
 - + $x^{(i)}$ ảnh train thứ i có kích thước $32 \times 32 \times 3$.
 - + $y^{(i)}$ nhãn ảnh thứ $i \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$.

Câu hỏi

- Mỗi điểm dữ liệu trong bộ dữ liệu Datatrain của CIFAR 10 có bao nhiêu thành phần?
 - Datatrain of CIFAR10:
$$\mathcal{D}_{train} = \{(x^{(i)}, y^{(i)})\}_{i=0}^{N-1}$$
 - Where:
 - + N : 50.000.
 - + $x^{(i)}$ ảnh train thứ i có kích thước $32 \times 32 \times 3$.
 - + $y^{(i)}$ nhãn ảnh thứ $i \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$.

Câu hỏi

- Mỗi điểm dữ liệu trong bộ dữ liệu Datatrain của CIFAR 10 có bao nhiêu thành phần?
 - Mỗi điểm dữ liệu trong bộ dữ liệu Datatrain của CIFAR 10 có hai thành phần là $(x^{(i)}, y^{(i)})$.
- Datatrain of CIFAR10:
- $$\mathcal{D}_{train} = \{(x^{(i)}, y^{(i)})\}_{i=0}^{N-1}$$
- Where:
 - + N : 50.000.
 - + $x^{(i)}$ ảnh train thứ i có kích thước $32 \times 32 \times 3$.
 - + $y^{(i)}$ nhãn ảnh thứ $i \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$.

Câu hỏi

- Thành phần thứ nhất của một điểm dữ liệu trong bộ dữ liệu Datatrain của CIFAR 10 có ký hiệu là gì? Được đọc như thế nào?
 - Datatrain of CIFAR10:
$$\mathcal{D}_{train} = \{(x^{(i)}, y^{(i)})\}_{i=0}^{N-1}$$
 - Where:
 - + N : 50.000.
 - + $x^{(i)}$ ảnh train thứ i có kích thước $32 \times 32 \times 3$.
 - + $y^{(i)}$ nhãn ảnh thứ $i \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$.

Câu hỏi

- Thành phần thứ nhất của một điểm dữ liệu trong bộ dữ liệu Datatrain của CIFAR 10 có ký hiệu là gì? Được đọc như thế nào?
- Thành phần thứ nhất của một điểm dữ liệu trong bộ dữ liệu Datatrain của CIFAR 10 có ký hiệu là $x^{(i)}$. Thành phần $x^{(i)}$ được đọc là ảnh train thứ i có kích thước $32 \times 32 \times 3$.
- **Datatrain of CIFAR10:**
 $\mathcal{D}_{train} = \{(x^{(i)}, y^{(i)})\}_{i=0}^{N-1}$
- Where:
 - + N : 50.000.
 - + $x^{(i)}$ ảnh train thứ i có kích thước $32 \times 32 \times 3$.
 - + $y^{(i)}$ nhãn ảnh thứ $i \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$.

Câu hỏi

- Thành phần thứ hai của một điểm dữ liệu trong bộ dữ liệu Datatrain của CIFAR 10 có ký hiệu là gì? Được đọc như thế nào?
 - Datatrain of CIFAR10:
$$\mathcal{D}_{train} = \{(x^{(i)}, y^{(i)})\}_{i=0}^{N-1}$$
 - Where:
 - + N : 50.000.
 - + $x^{(i)}$ ảnh train thứ i có kích thước $32 \times 32 \times 3$.
 - + $y^{(i)}$ nhãn ảnh thứ $i \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$.

Câu hỏi

- Thành phần thứ hai của một điểm dữ liệu trong bộ dữ liệu Datatrain của CIFAR 10 có ký hiệu là gì? Được đọc như thế nào?
- Thành phần thứ hai của một điểm dữ liệu trong bộ dữ liệu Datatrain của CIFAR 10 có ký hiệu là $y^{(i)}$. Thành phần $y^{(i)}$ được đọc là nhãn của ảnh train thứ i là một giá trị thuộc tập hợp $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$.
- Datatrain of CIFAR10:

$$\mathcal{D}_{train} = \{(x^{(i)}, y^{(i)})\}_{i=0}^{N-1}$$
 - Where:
 - + N : 50.000.
 - + $x^{(i)}$ ảnh train thứ i có kích thước $32 \times 32 \times 3$.
 - + $y^{(i)}$ nhãn ảnh thứ $i \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$.

Câu hỏi

- Phép nhân $32 \times 32 \times 3$ cho kết quả bao nhiêu?
 - Datatrain of CIFAR10:
$$\mathcal{D}_{train} = \{(x^{(i)}, y^{(i)})\}_{i=0}^{N-1}$$
 - Where:
 - + N : 50.000.
 - + $x^{(i)}$ ảnh train thứ i có kích thước $32 \times 32 \times 3$.
 - + $y^{(i)}$ nhãn ảnh thứ $i \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$.

Câu hỏi

- Phép nhân $32 \times 32 \times 3$ cho kết quả bao nhiêu?
- Phép nhân $32 \times 32 \times 3$ cho kết quả là 3.072
- Datatrain of CIFAR10:
$$\mathcal{D}_{train} = \{(x^{(i)}, y^{(i)})\}_{i=0}^{N-1}$$
 - Where:
 - + N : 50.000.
 - + $x^{(i)}$ ảnh train thứ i có kích thước $32 \times 32 \times 3$.
 - + $y^{(i)}$ nhãn ảnh thứ $i \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$.

Câu hỏi

— Bộ dữ liệu Datatest của CIFAR 10 có bao nhiêu điểm dữ liệu?

— Datatest of CIFAR10:

$$\mathcal{D}_{test} = \{(x^{(i)}, y^{(i)})\}_{i=0}^{M-1}$$

— Where:

+ M : 10.000.

+ $x^{(i)}$ ảnh test thứ i có kích thước $32 \times 32 \times 3$.

+ $y^{(i)}$ nhãn ảnh thứ $i \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$.

Câu hỏi

- Bộ dữ liệu Datatest của CIFAR 10 có bao nhiêu điểm dữ liệu?
- Bộ dữ liệu Datatest của CIFAR 10 có 10.000 điểm dữ liệu.

- Datatest of CIFAR10:
$$\mathcal{D}_{test} = \{(x^{(i)}, y^{(i)})\}_{i=0}^{M-1}$$
- Where:
 - + M : 10.000.
 - + $x^{(i)}$ ảnh test thứ i có kích thước $32 \times 32 \times 3$.
 - + $y^{(i)}$ nhãn ảnh thứ $i \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$.

Câu hỏi

— Bộ dữ liệu Datatest của CIFAR 10 có kích thước bao nhiêu?

— Datatest of CIFAR10:

$$\mathcal{D}_{test} = \{(x^{(i)}, y^{(i)})\}_{i=0}^{M-1}$$

— Where:

+ M : 10.000.

+ $x^{(i)}$ ảnh test thứ i có kích thước $32 \times 32 \times 3$.

+ $y^{(i)}$ nhãn ảnh thứ $i \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$.

Câu hỏi

- Bộ dữ liệu Datatest của CIFAR 10 có kích thước bao nhiêu?
- Bộ dữ liệu Datatest của CIFAR 10 có kích thước 10.000

- Datatest of CIFAR10:
$$\mathcal{D}_{test} = \{(x^{(i)}, y^{(i)})\}_{i=0}^{M-1}$$
- Where:
 - + M : 10.000.
 - + $x^{(i)}$ ảnh test thứ i có kích thước $32 \times 32 \times 3$.
 - + $y^{(i)}$ nhãn ảnh thứ $i \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$.

BÀI TẬP CIFAR 10

Bài tập CIFAR 10

- Bài tập 01:
 - + Load bộ dữ liệu CIFAR10 theo nguyên bản của đại học Toronto về thư mục làm việc của Colab.

Bài tập CIFAR 10

- Bài tập 02:
 - + Load bộ dữ liệu CIFAR10 trong thư mục làm việc của Colab.
 - + Đếm số lượng điểm dữ liệu theo từng lớp nhãn trong bộ dữ liệu train nguyên bản.
 - + Đếm số lượng điểm dữ liệu theo từng lớp nhãn trong bộ dữ liệu test nguyên bản.
 - + Vẽ biểu đồ minh họa số lượng điểm dữ liệu theo từng lớp nhãn trong bộ dữ liệu (train và test).

Bài tập CIFAR 10

— Bài tập 03:

- + Load bộ dữ liệu CIFAR10 trong thư mục làm việc của Colab.
- + Chia bộ dữ liệu train có kích thước 50000 điểm dữ liệu thành hai bộ dữ liệu con có tên.
 - Bộ dữ liệu uitTrainCIFAR10 có kích thước 40000 điểm dữ liệu (số lượng điểm dữ liệu theo từng lớp nhãn bằng nhau).
 - Bộ dữ liệu uitValidationCIFAR10 có kích thước 10000.
- + Bộ dữ liệu test uitTestCIFAR10 có kích thước 10000 điểm dữ liệu (giống theo nguyên bản).

Bài tập CIFAR 10

— Bài tập 04:

- + Load bộ dữ liệu uitTrainCIFAR10 trong thư mục làm việc của Colab.
- + Hiển thị 10 ảnh đầu tiên theo từng lớp nhãn.

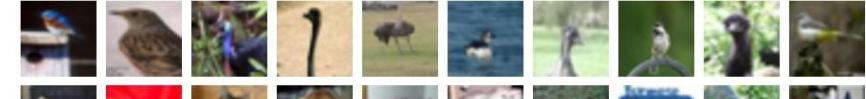
airplane



automobile



bird



cat



deer



dog



frog



horse



ship



truck



Bài tập CIFAR 10

— Bài tập 05A:

- + Load bộ dữ liệu uitTrainCIFAR10 trong thư mục làm việc của Colab.
- + Tạo bộ dữ liệu uitTinyTrainCIFAR10 với mỗi lớp nhãn có 10 ảnh được lấy từ bộ dữ liệu uitTrainCIFAR10.

airplane



automobile



bird



cat



deer



dog



frog



horse



ship



truck

Bài tập CIFAR 10

— Bài tập 05B:

+ Load bộ dữ liệu uitValidationCIFAR10 trong thư mục làm việc của Colab.

+ Tạo bộ dữ liệu uitTinyValidationCIFAR10 với mỗi lớp nhãn có 3 ảnh được lấy từ bộ dữ liệu uitValidationCIFAR10.

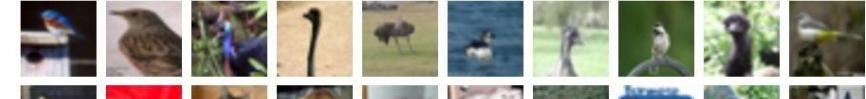
airplane



automobile



bird



cat



deer



dog



frog



horse



ship



truck



Bài tập CIFAR 10

— Bài tập 05C:

- + Load bộ dữ liệu uitTestCIFAR10 trong thư mục làm việc của Colab.
- + Tạo bộ dữ liệu uitTinyTestCIFAR10 với mỗi lớp nhãn có 3 ảnh được lấy từ bộ dữ liệu uitTestCIFAR10.

airplane



automobile



bird



cat



deer



dog



frog



horse



ship



truck



Bài tập CIFAR 10

— Bài tập 06A:

- + Load bộ dữ liệu uitTrainCIFAR10 trong thư mục làm việc của Colab.
- + Tạo bộ dữ liệu uitSmallTrainCIFAR10 với mỗi lớp nhãn có 100 ảnh được lấy từ bộ dữ liệu uitTrainCIFAR10.

airplane



automobile



bird



cat



deer



dog



frog



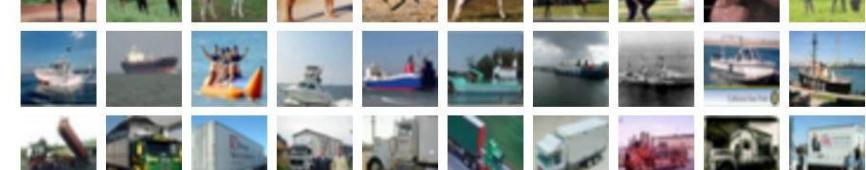
horse



ship



truck



Bài tập CIFAR 10

— Bài tập 06B:

+ Load bộ dữ liệu uitValidationCIFAR10 trong thư mục làm việc của Colab.

+ Tạo bộ dữ liệu uitSmallValidationCIFAR10 với mỗi lớp nhãn có 25 ảnh được lấy từ bộ dữ liệu uitValidationCIFAR10.

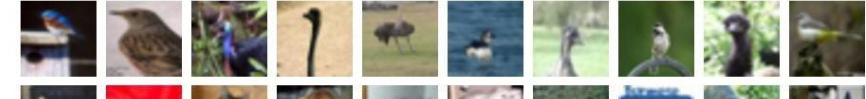
airplane



automobile



bird



cat



deer



dog



frog



horse



ship



truck

Bài tập CIFAR 10

— Bài tập 06C:

- + Load bộ dữ liệu uitTestCIFAR10 trong thư mục làm việc của Colab.
- + Tạo bộ dữ liệu uitSmallTestCIFAR10 với mỗi lớp nhãn có 25 ảnh được lấy từ bộ dữ liệu uitTestCIFAR10.

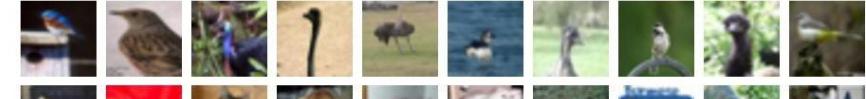
airplane



automobile



bird



cat



deer



dog



frog



horse



ship



truck



Bài tập CIFAR 10

— Bài tập 07A:

- + Load bộ dữ liệu uitTrainCIFAR10 trong thư mục làm việc của Colab.
- + Tạo bộ dữ liệu uitMediumTrainCIFAR10 với mỗi lớp nhãn có 1000 ảnh được lấy từ bộ dữ liệu uitTrainCIFAR10.

airplane



automobile



bird



cat



deer



dog



frog



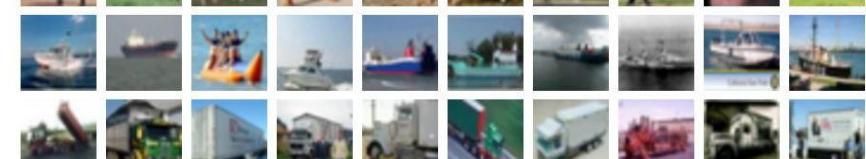
horse



ship



truck



Bài tập CIFAR 10

— Bài tập 07B:

+ Load bộ dữ liệu uitValidationCIFAR10 trong thư mục làm việc của Colab.

+ Tạo bộ dữ liệu uitMediumValidationCIFAR10 với mỗi lớp nhãn có 250 ảnh được lấy từ bộ dữ liệu uitValidationCIFAR10.

airplane



automobile



bird



cat



deer



dog



frog



horse



ship



truck



Bài tập CIFAR 10

— Bài tập 07C:

- + Load bộ dữ liệu uitTestCIFAR10 trong thư mục làm việc của Colab.
- + Tạo bộ dữ liệu uitMediumTestCIFAR10 với mỗi lớp nhãn có 250 ảnh được lấy từ bộ dữ liệu uitTestCIFAR10.

airplane



automobile



bird



cat



deer



dog



frog



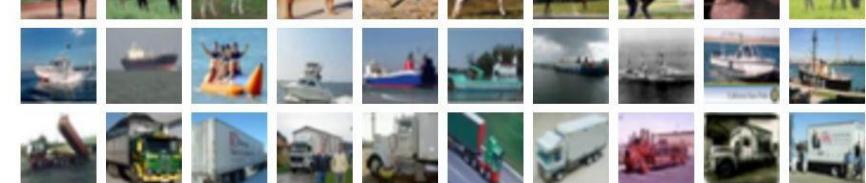
horse



ship



truck



Bài tập CIFAR 10

— Bài tập 08A:

- + Load bộ dữ liệu uitTrainCIFAR10 trong thư mục làm việc của Colab.
- + Tạo bộ dữ liệu uitHalfTrainCIFAR10 với mỗi lớp nhãn có 2000 ảnh được lấy từ bộ dữ liệu uitTrainCIFAR10.

airplane



automobile



bird



cat



deer



dog



frog



horse



ship



truck



Bài tập CIFAR 10

— Bài tập 08B:

+ Load bộ dữ liệu uitValidationCIFAR10 trong thư mục làm việc của Colab.

+ Tạo bộ dữ liệu uitHalfValidationCIFAR10 với mỗi lớp nhãn có 500 ảnh được lấy từ bộ dữ liệu uitValidationCIFAR10.

airplane



automobile



bird



cat



deer



dog



frog



horse



ship



truck



Bài tập CIFAR 10

— Bài tập 08C:

- + Load bộ dữ liệu uitTestCIFAR10 trong thư mục làm việc của Colab.
- + Tạo bộ dữ liệu uitHalfTestCIFAR10 với mỗi lớp nhãn có 500 ảnh được lấy từ bộ dữ liệu uitTestCIFAR10.

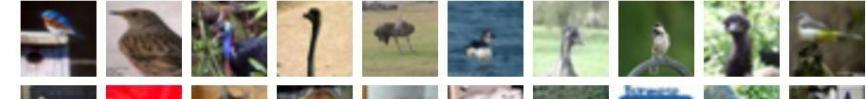
airplane



automobile



bird



cat



deer



dog



frog



horse



ship



truck



First classifier

NEAREST NEIGHBOR

Data – Driven Approach

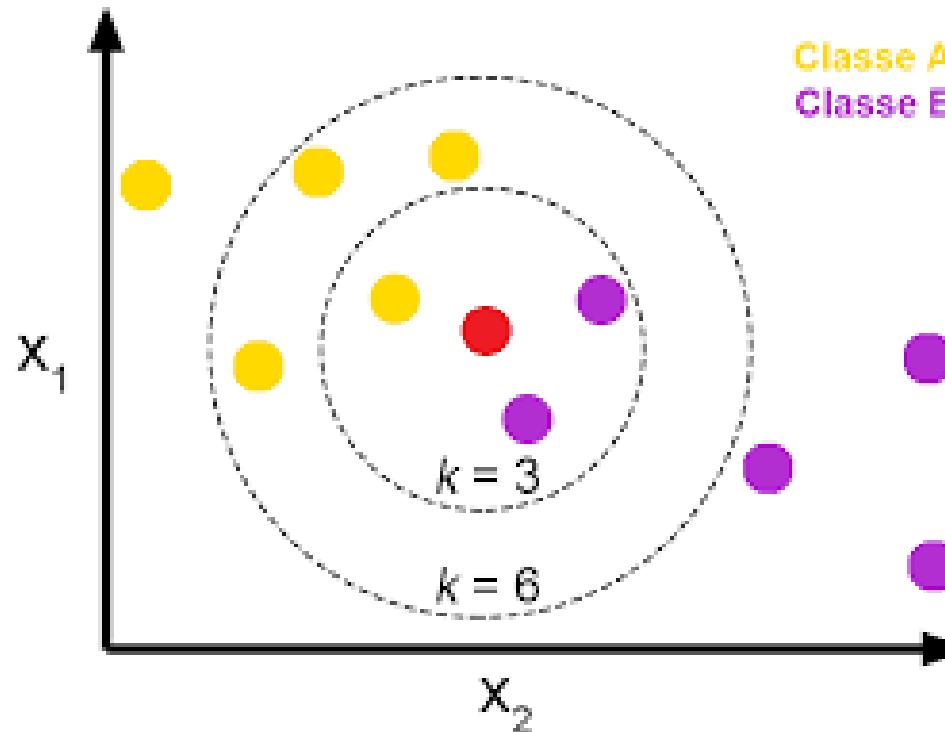
1. Collect a dataset of images and labels
2. Use Machine Learning to train a classifier
3. Evaluate the classifier on new images

```
def train(images, labels):  
    # Machine learning!  
    return model
```

```
def predict(model, test_images):  
    # Use model to predict labels  
    return test_labels
```

Nearest Neighbor Classifier

- Assign label of nearest training data point to each test data point.
- Gán nhãn điểm dữ liệu huấn luyện gần nhất cho từng điểm dữ liệu kiểm tra.



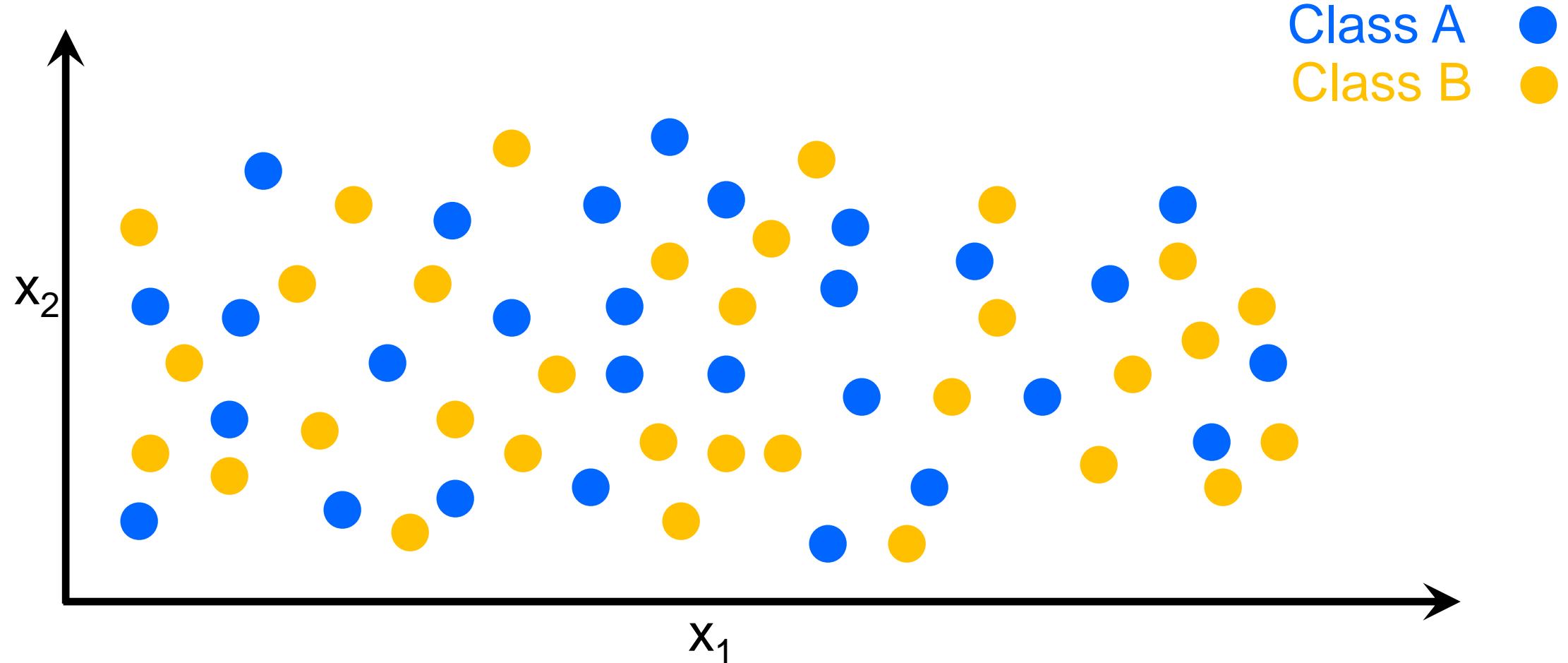
Nearest Neighbor Classifier

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

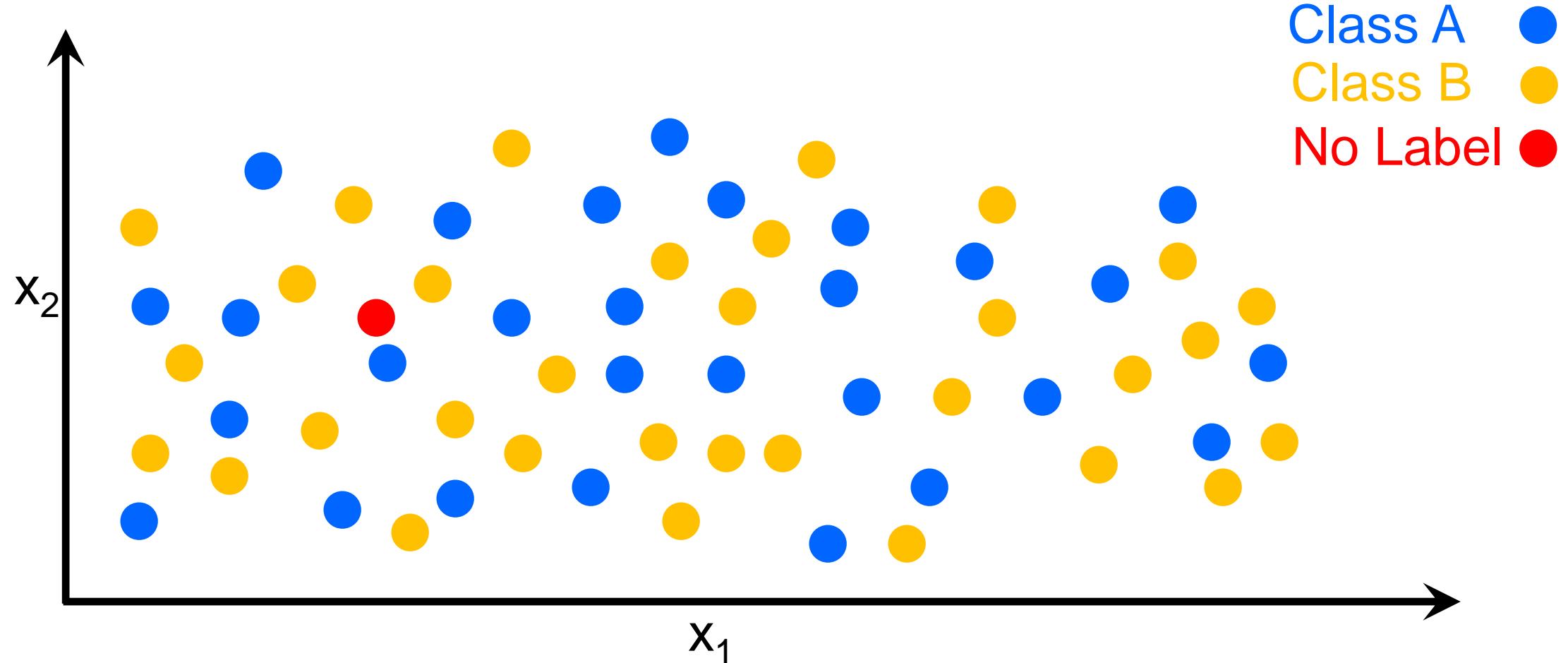
- All we need is a distance function for our inputs.
- No training required!

Nearest neighbor **EXAMPLE K-NN**

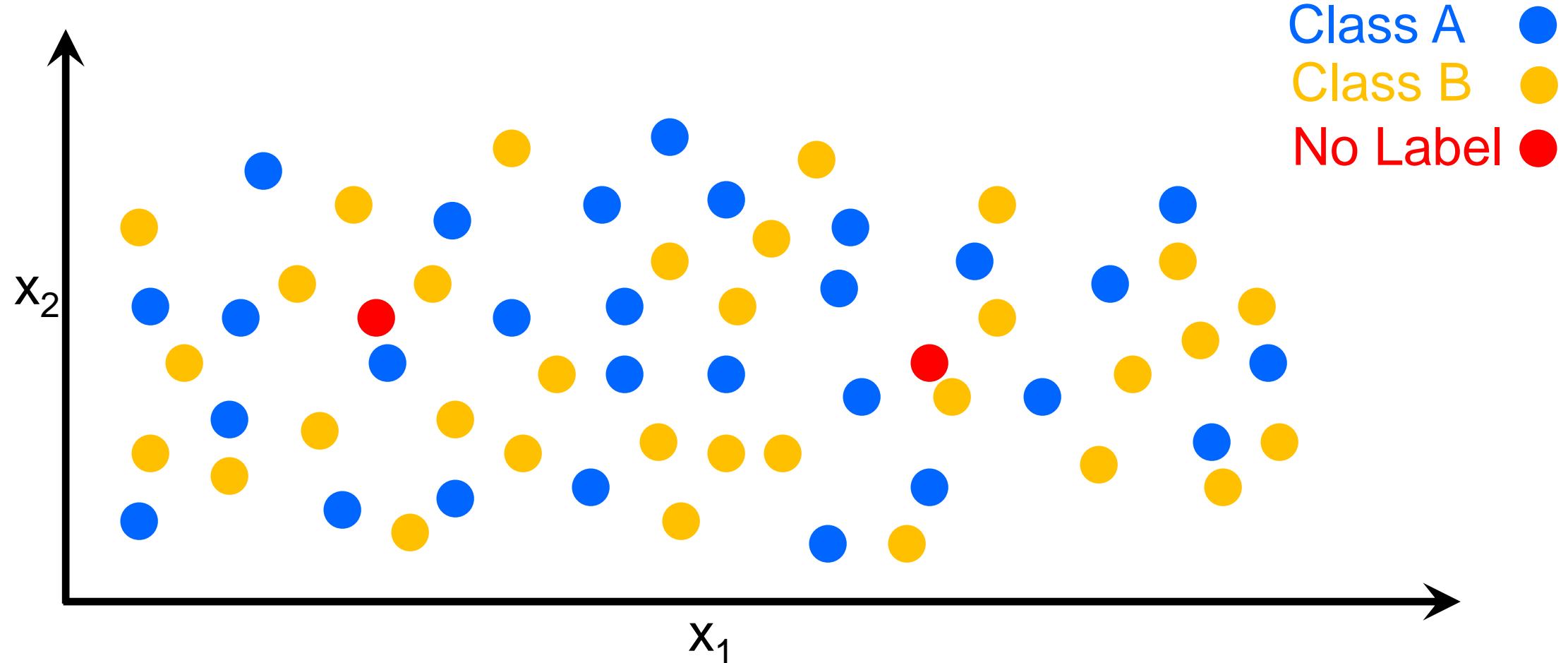
Nearest Neighbor Classifier



Nearest Neighbor Classifier



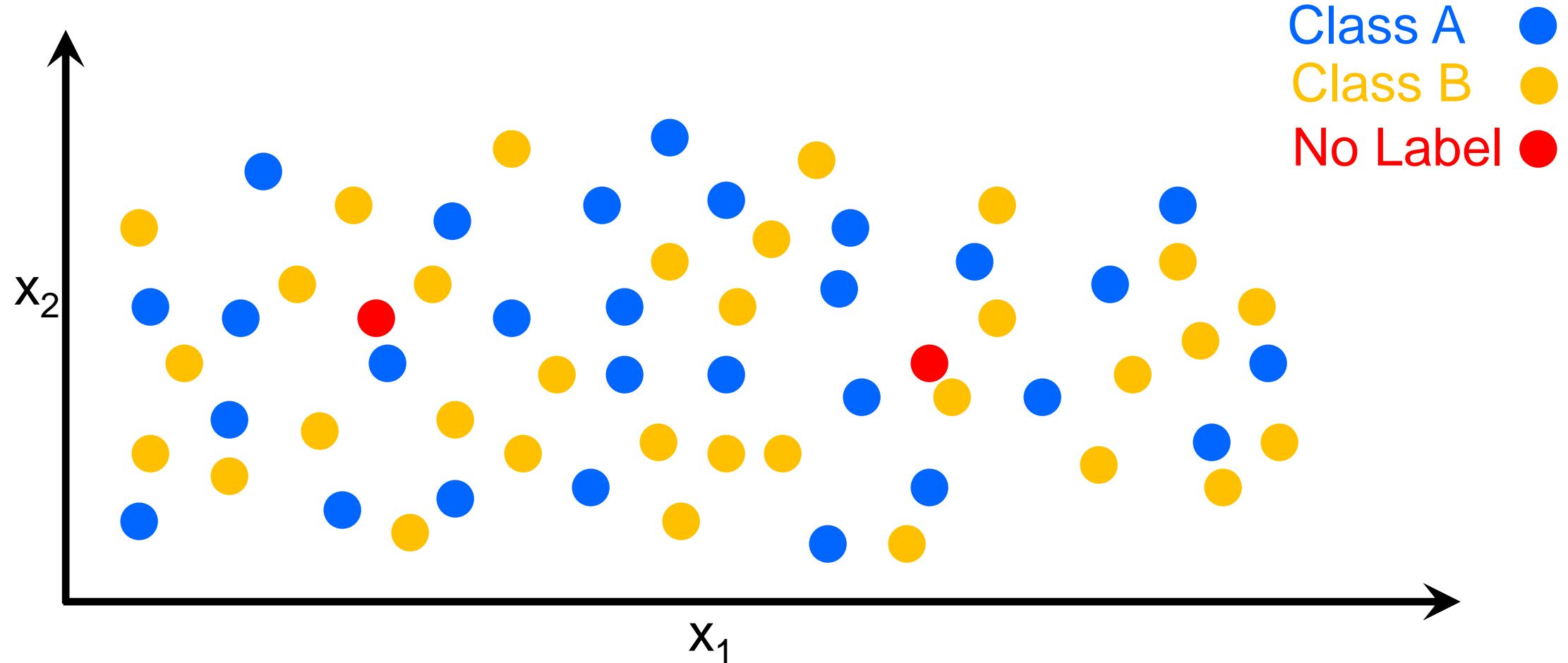
Nearest Neighbor Classifier



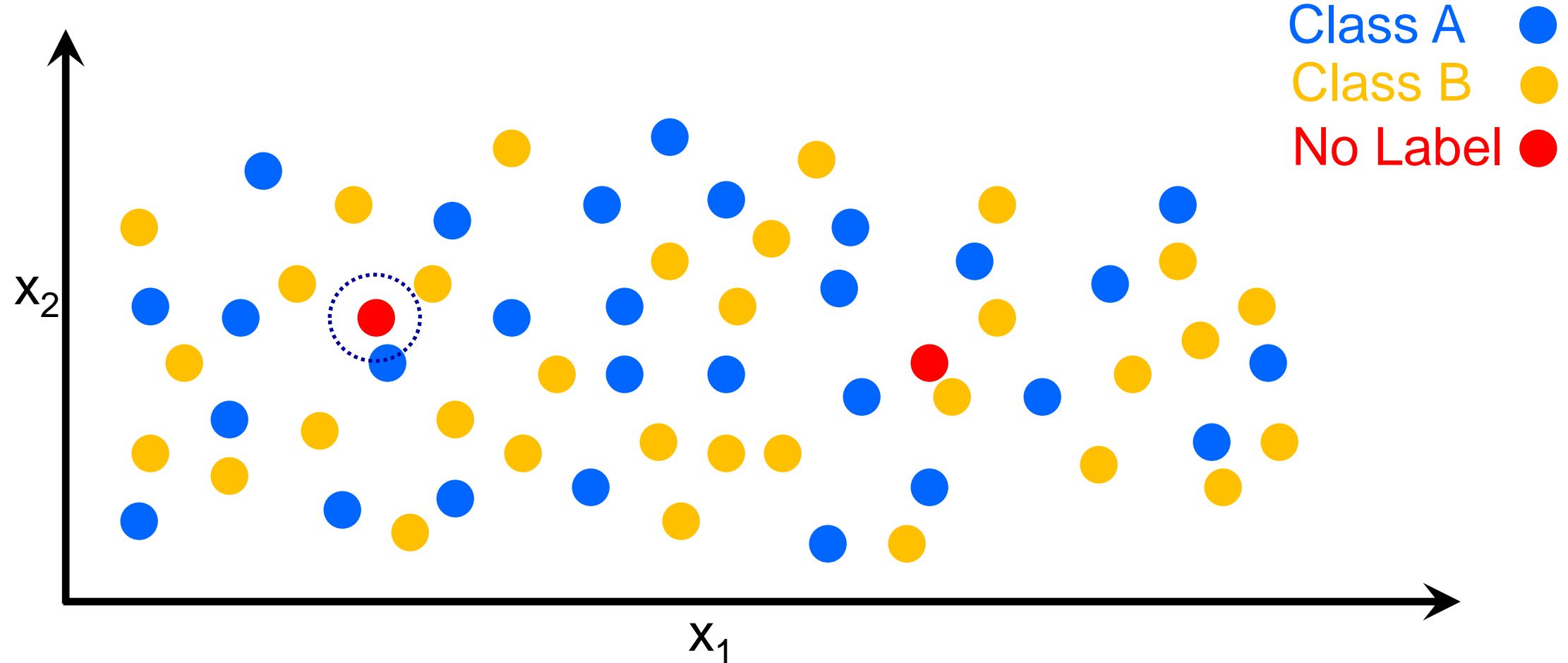
Nearest neighbor

1 – NEAREST NEIGHBOR

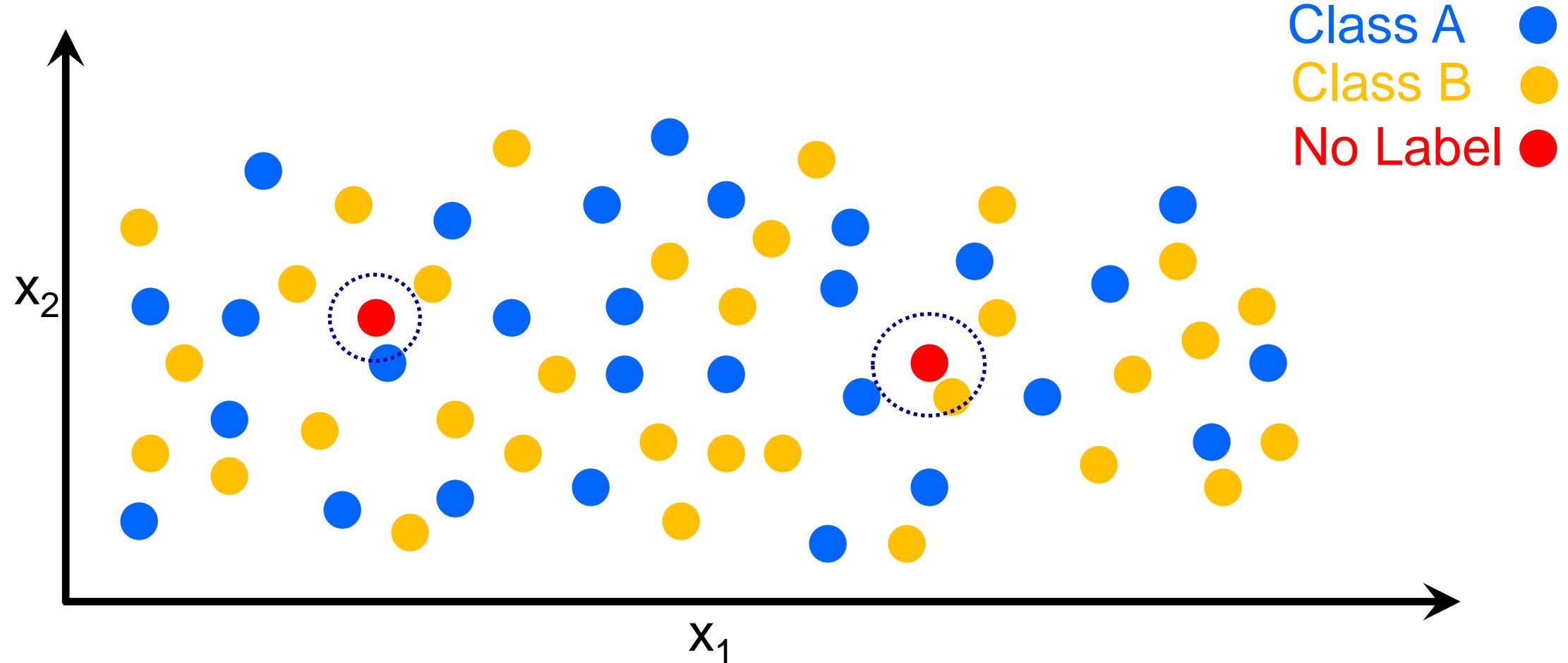
1 – Nearest neighbor



1 – Nearest neighbor



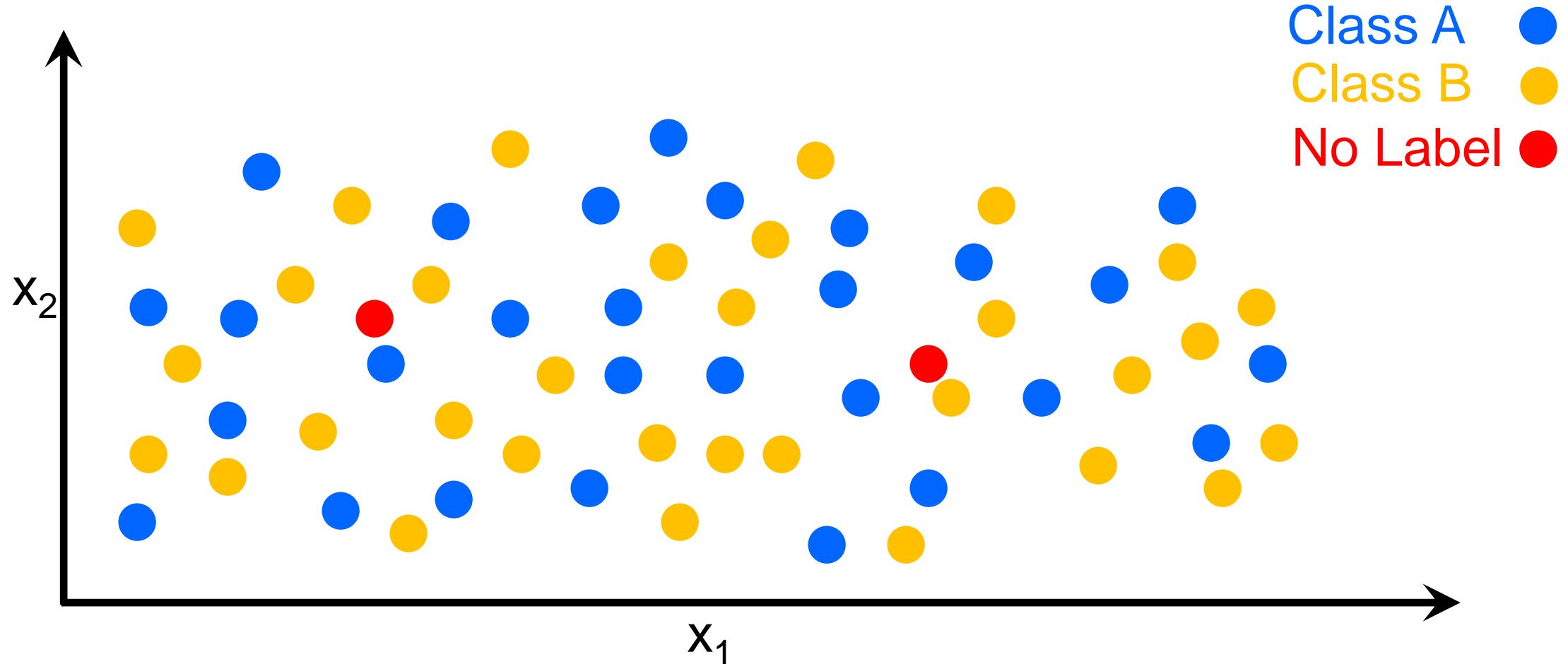
1 – Nearest neighbor



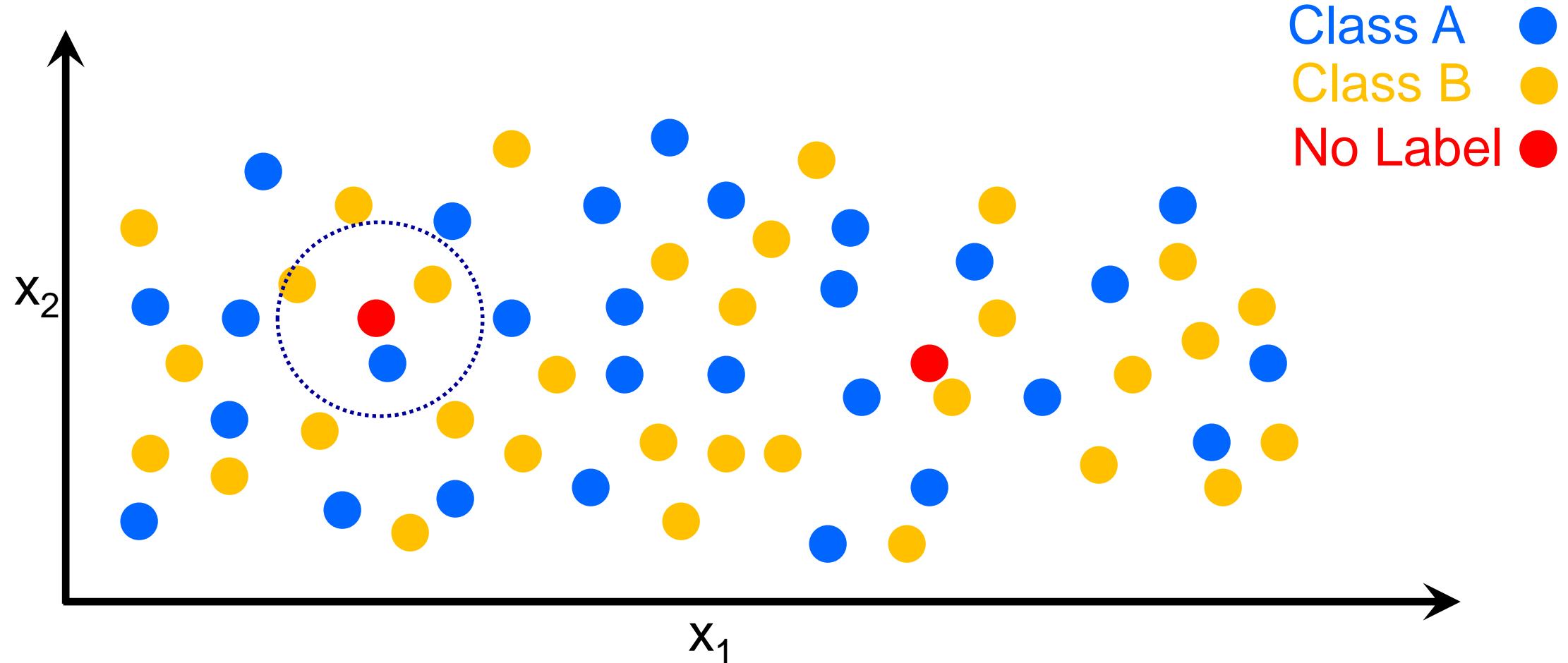
Nearest neighbor

3 – NEAREST NEIGHBOR

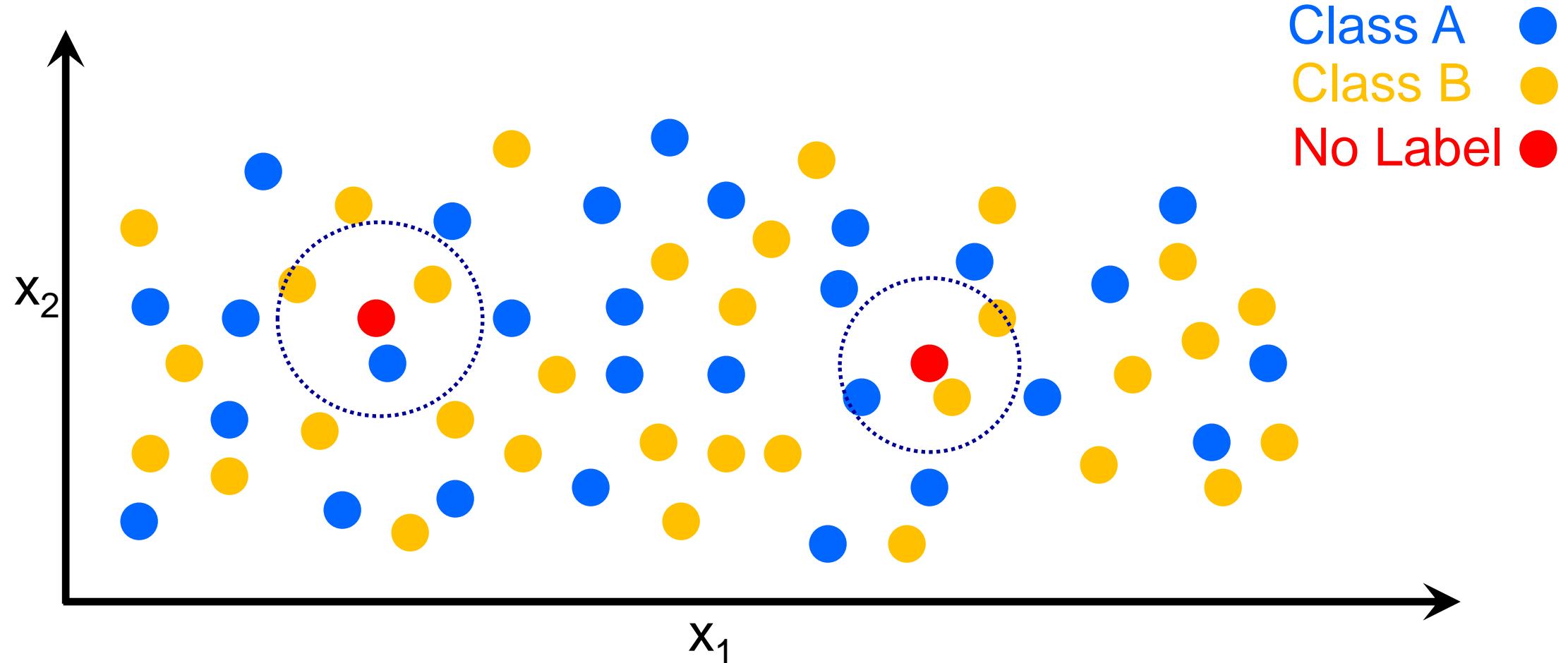
3 – Nearest neighbor



3 – Nearest neighbor



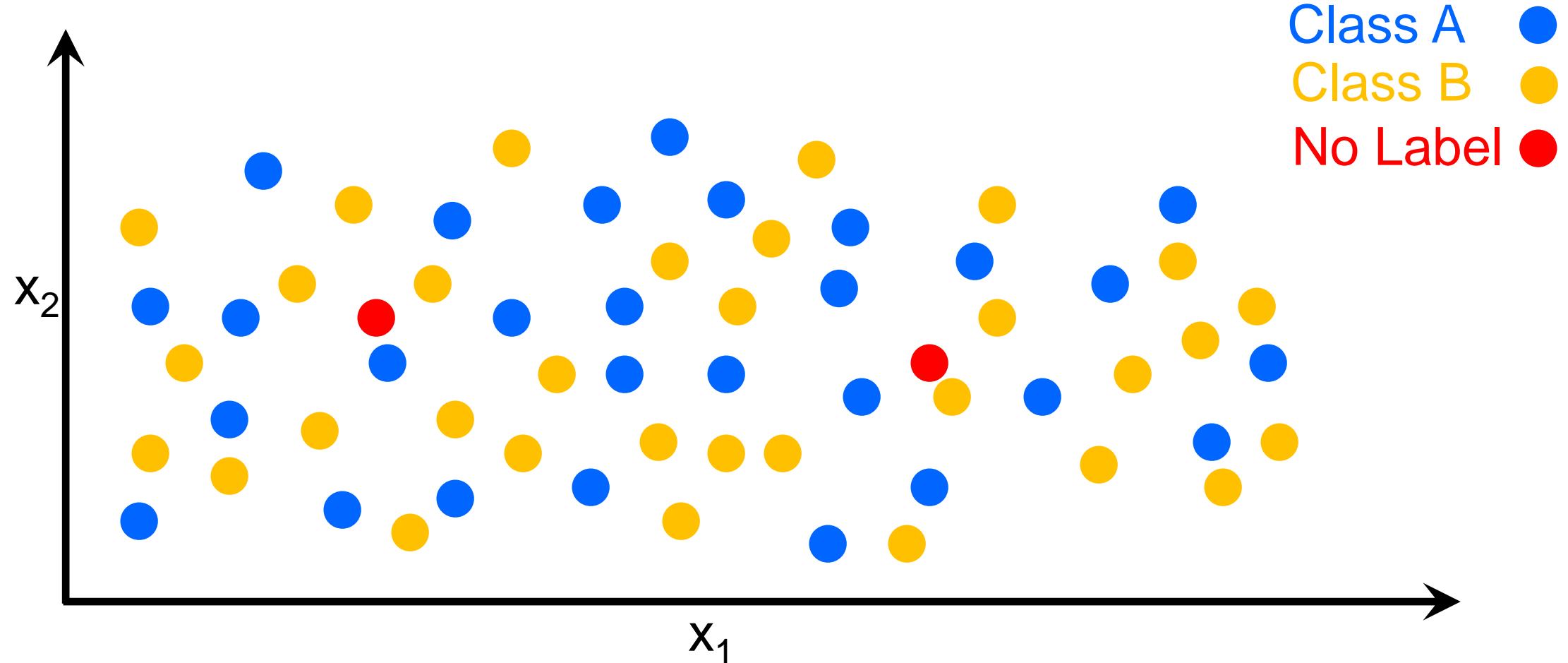
3 – Nearest neighbor



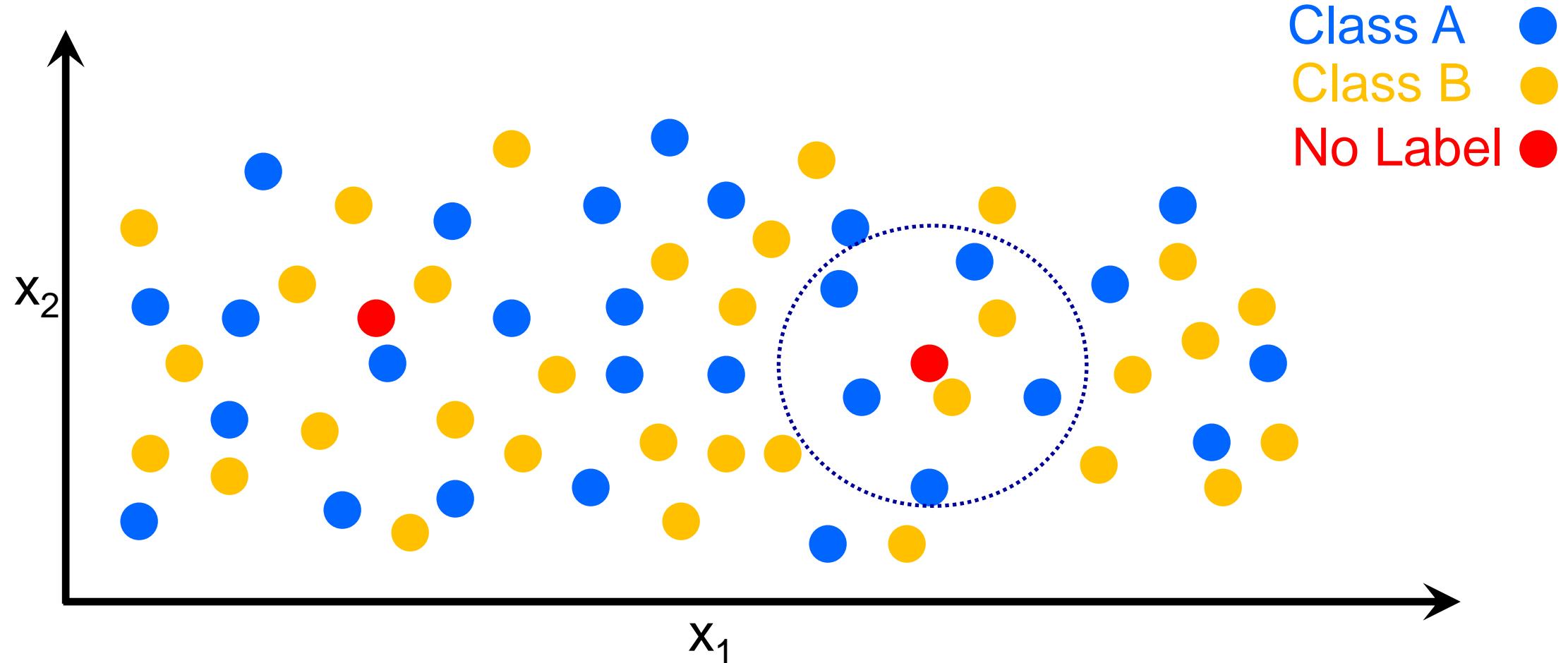
Nearest neighbor

7 – NEAREST NEIGHBOR

7 – Nearest neighbor



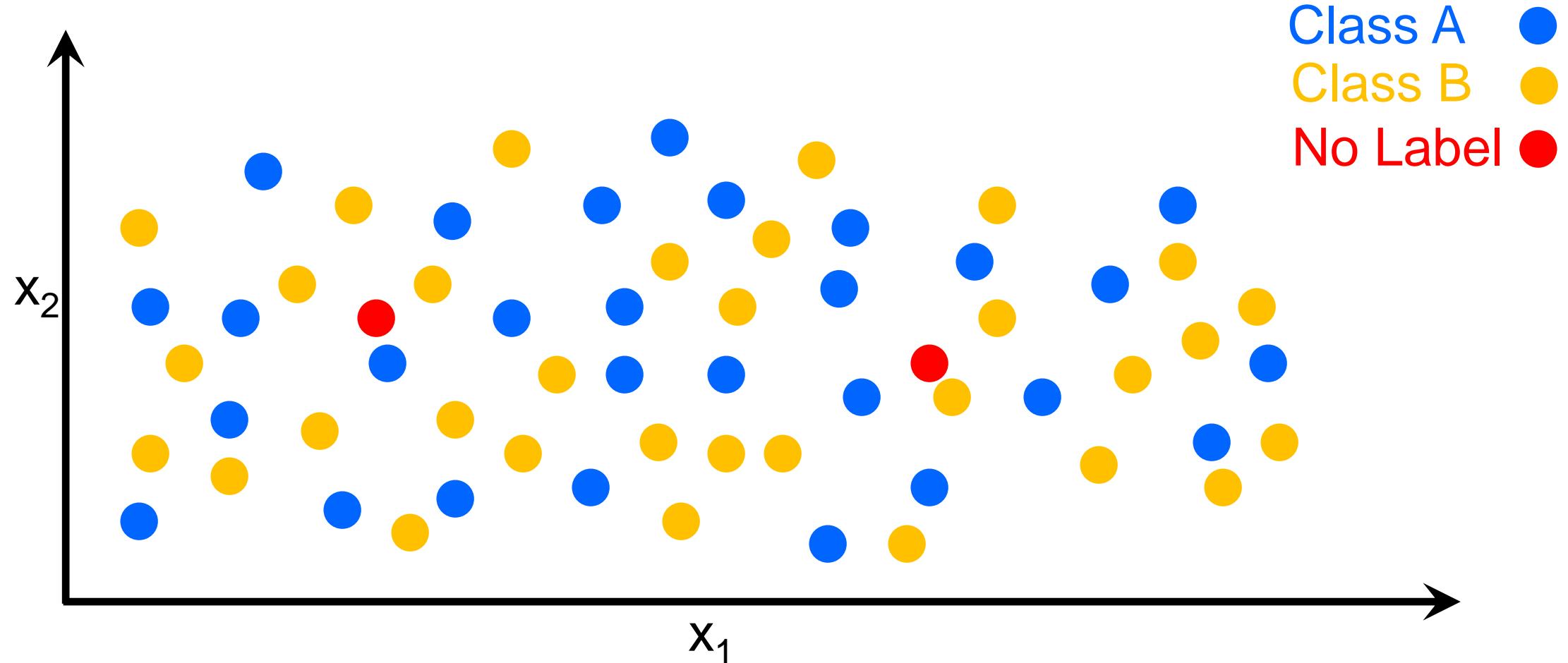
7 – Nearest neighbor



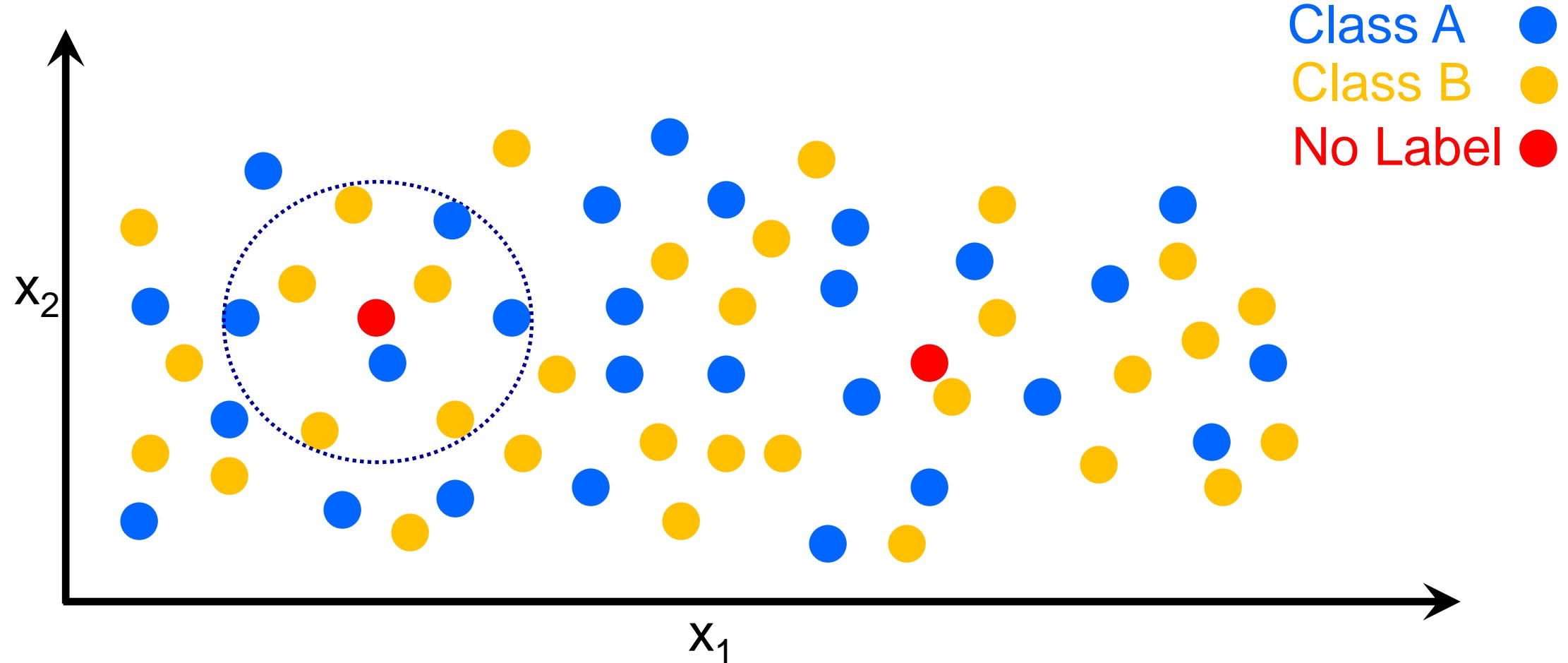
Nearest neighbor

9 – NEAREST NEIGHBOR

9 – Nearest neighbor



9 – Nearest neighbor



Nearest neighbor

NEAREST NEIGHBOR CLASSIFIER

Nearest Neighbor Classifier

```
def train(images, labels):  
    # Machine learning!  
    return model
```

→ Memorize all
data train and
labels

```
def predict(model, test_images):  
    # Use model to predict labels  
    return test_labels
```

→ Predict the label of
the most similar
training image

Nearest neighbor
DISTANCE METRIC

Distance Metric

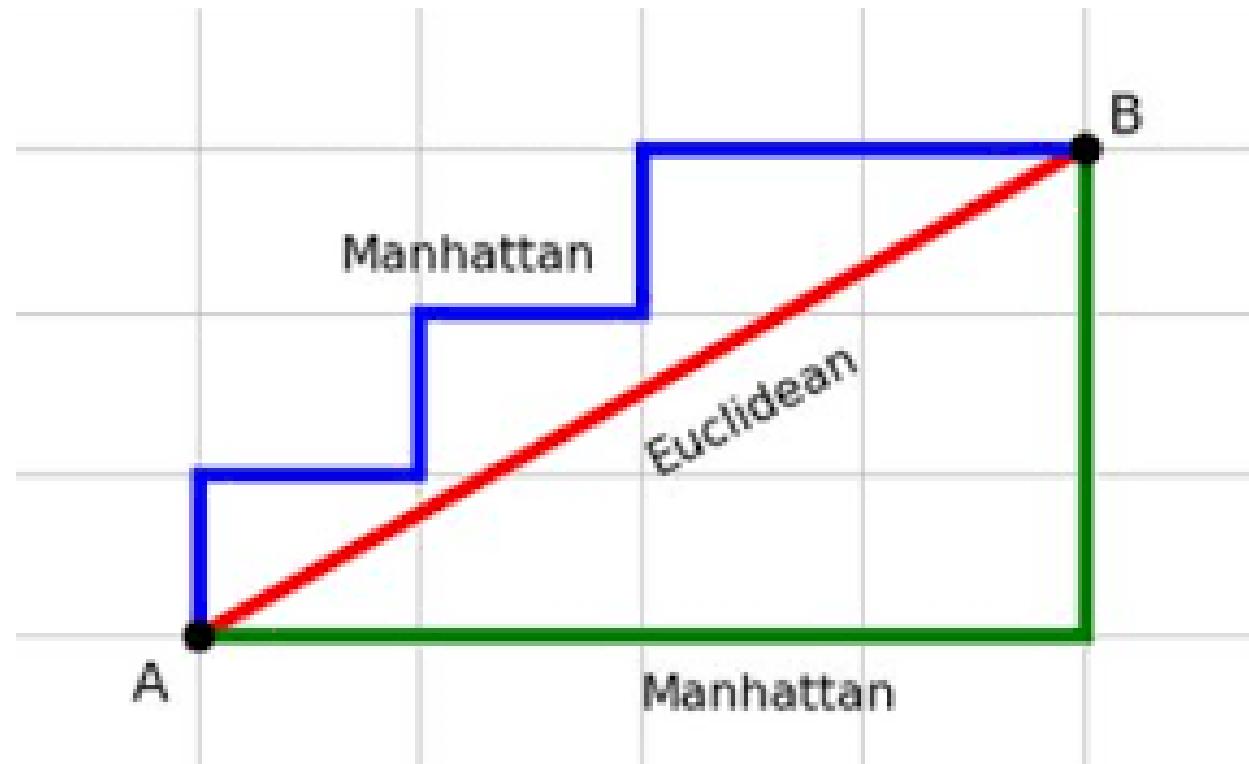
— L1 distance:

$$d_1(I_1, I_2) = \sum_p |I_1^p - I_2^p|$$

| test image | | | | training image | | | | pixel-wise absolute value differences | | | |
|------------|----|-----|-----|----------------|----|-----|-----|---------------------------------------|----|----|-----|
| 56 | 32 | 10 | 18 | 10 | 20 | 24 | 17 | 46 | 12 | 14 | 1 |
| 90 | 23 | 128 | 133 | 8 | 10 | 89 | 100 | 82 | 13 | 39 | 33 |
| 24 | 26 | 178 | 200 | 12 | 16 | 178 | 170 | 12 | 10 | 0 | 30 |
| 2 | 0 | 255 | 220 | 4 | 32 | 233 | 112 | 2 | 32 | 22 | 108 |

- = → 456

Distance Metric



Nearest neighbor **IMPLEMENTATION**

Nearest Neighbor classifier

Memorize training data

```
import numpy as np

class NearestNeighbor:
    def __init__(self):
        pass

    def train(self, X, y):
        """
        X is N x D where each row is an example. Y is 1-dimension of size N """
        # the nearest neighbor classifier simply remembers all the training data
        self.Xtr = X
        self.ytr = y

    def predict(self, X):
        """
        X is N x D where each row is an example we wish to predict label for """
        num_test = X.shape[0]
        # lets make sure that the output type matches the input type
        Ypred = np.zeros(num_test, dtype = self.ytr.dtype)

        # loop over all test rows
        for i in xrange(num_test):
            # find the nearest training image to the i'th test image
            # using the L1 distance (sum of absolute value differences)
            distances = np.sum(np.abs(self.Xtr - X[i,:]), axis = 1)
            min_index = np.argmin(distances) # get the index with smallest distance
            Ypred[i] = self.ytr[min_index] # predict the label of the nearest example

        return Ypred
```

Nearest Neighbor classifier

For each test image:
Find closest train image
predict label of nearest image

```
import numpy as np

class NearestNeighbor:
    def __init__(self):
        pass

    def train(self, X, y):
        """
        X is N x D where each row is an example. Y is 1-dimension of size N """
        # the nearest neighbor classifier simply remembers all the training data
        self.Xtr = X
        self.ytr = y

    def predict(self, X):
        """
        X is N x D where each row is an example we wish to predict label for """
        num_test = X.shape[0]
        # lets make sure that the output type matches the input type
        Ypred = np.zeros(num_test, dtype = self.ytr.dtype)

        # loop over all test rows
        for i in xrange(num_test):
            # find the nearest training image to the i'th test image
            # using the L1 distance (sum of absolute value differences)
            distances = np.sum(np.abs(self.Xtr - X[i,:]), axis = 1)
            min_index = np.argmin(distances) # get the index with smallest distance
            Ypred[i] = self.ytr[min_index] # predict the label of the nearest example

        return Ypred
```

Nearest neighbor **DISCUSSION**

Nearest Neighbor Classifier

- Question: With N examples, how fast are training and prediction?

Nearest Neighbor Classifier

- Question: With N examples, how fast are training and prediction?
- Answer:

Nearest Neighbor Classifier

- Question: With N examples, how fast are training and prediction?
- Answer:
 - + Train: $O(1)$.

Nearest Neighbor Classifier

- Question: With N examples, how fast are training and prediction?
- Answer:
 - + Train: $O(1)$.
 - + Predict: $O(N)$.

Nearest Neighbor Classifier

- Question: With N examples, how fast are training and prediction?
- Answer:
 - + Train: $O(1)$.
 - + Predict: $O(N)$.
- This is bad:

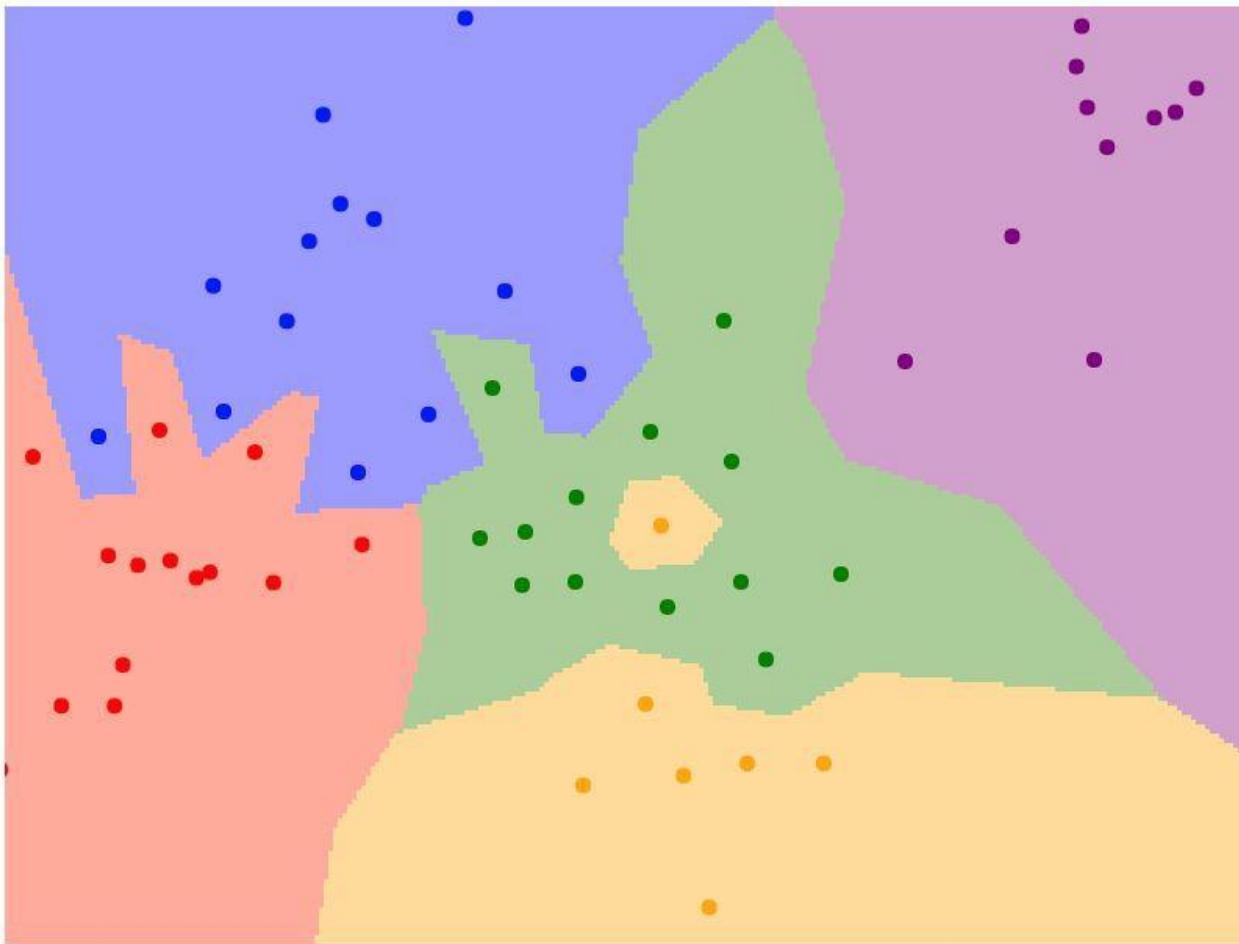
Nearest Neighbor Classifier

- Question: With N examples, how fast are training and prediction?
- Answer:
 - + Train: $O(1)$.
 - + Predict: $O(N)$.
- This is bad:
 - + we want classifiers that are fast at prediction;

Nearest Neighbor Classifier

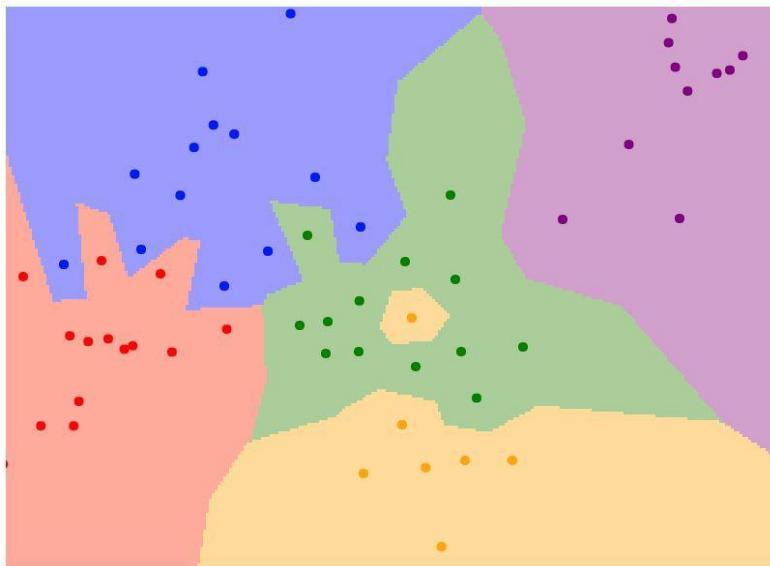
- Question: With N examples, how fast are training and prediction?
- Answer:
 - + Train: $O(1)$.
 - + Predict: $O(N)$.
- This is bad:
 - + we want classifiers that are fast at prediction;
 - + slow for training is ok

What does this look like?

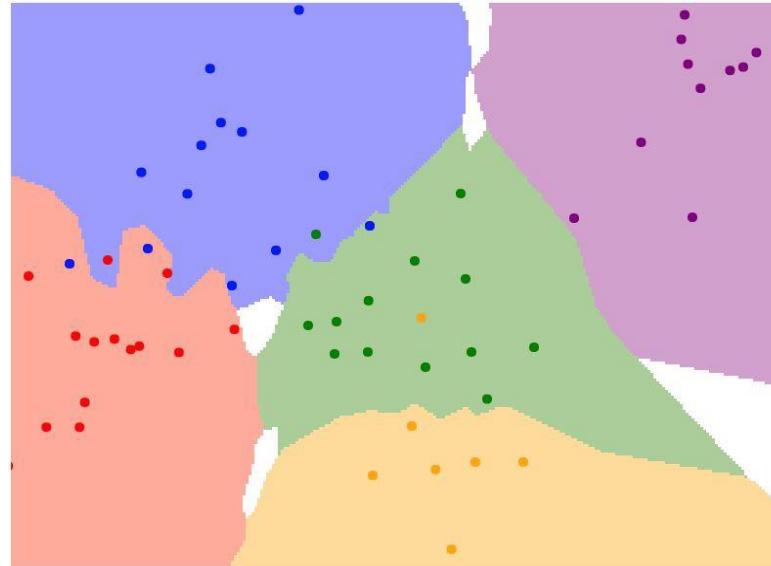


K-Nearest Neighbors

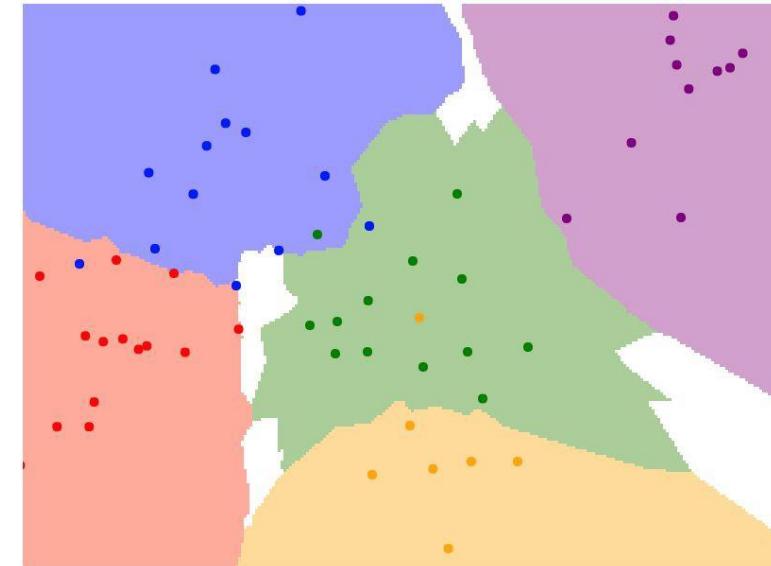
- Instead of copying label from nearest neighbor, take majority vote from K closest points.



$K = 1$



$K = 2$



$K = 3$

What does this look like?



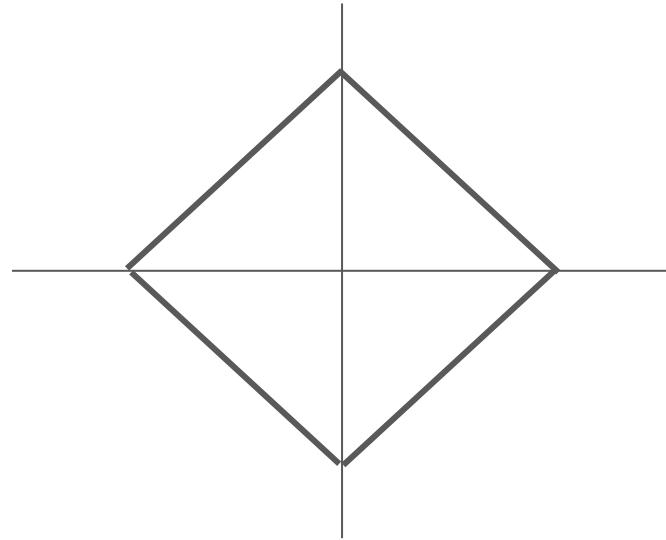
Nearest neighbor

DISTANCE METRIC CONTINUE

K-Nearest Neighbors: Distance Metric

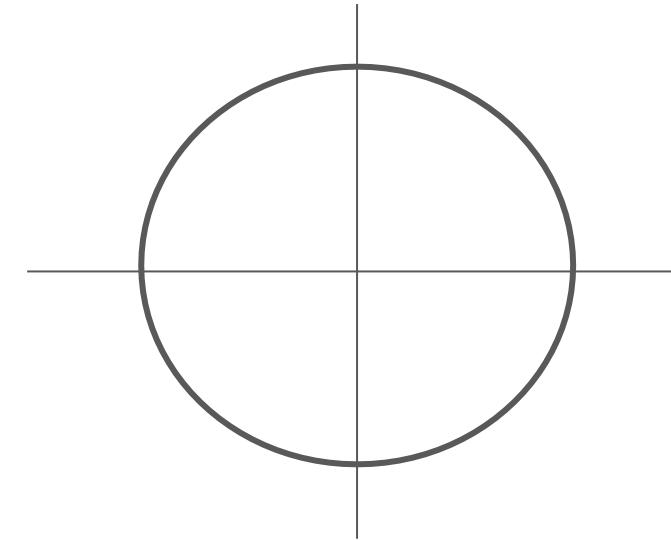
– L1 (Manhattan) distance

$$d_1(I_1, I_2) = \sum_p |I_1^p - I_2^p|$$



– L2 (Euclidean) distance

$$d_2(I_1, I_2) = \sqrt{\sum_p (I_1^p - I_2^p)^2}$$



K-Nearest Neighbors: Distance Metric

– L1 (Manhattan) distance

$$d_1(I_1, I_2) = \sum_p |I_1^p - I_2^p|$$



$K = 1$

– L2 (Euclidean) distance

$$d_1(I_1, I_2) = \sqrt{\sum_p (I_1^p - I_2^p)^2}$$



$K = 1$

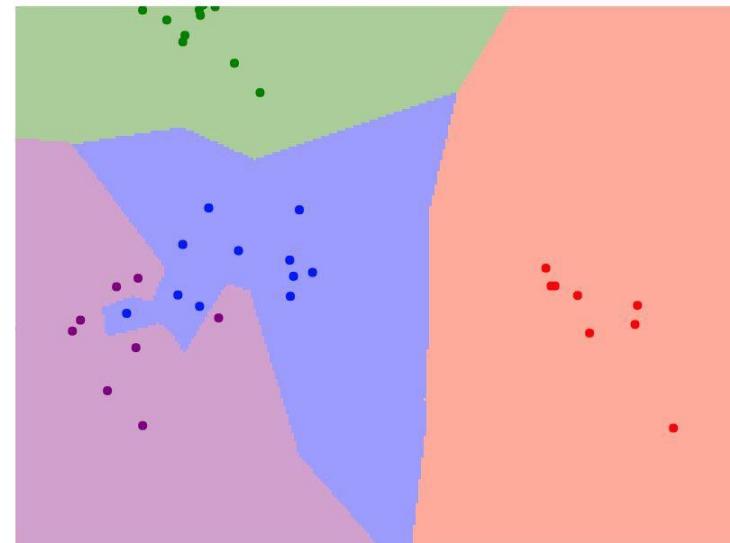
Nearest neighbor **DEMO**

K-Nearest Neighbors: Demo Time

K-Nearest Neighbors Demo

This interactive demo lets you explore the K-Nearest Neighbors algorithm for classification. Each point in the plane is colored with the class that would be assigned to it using the K-Nearest Neighbors algorithm. Points for which the K-Nearest Neighbor algorithm results in a tie are colored white.

You can move points around by clicking and dragging!



Metric

L1 L2

Num classes

2 3 4 5

Num Neighbors (K)

1 2 3 4 5 6 7

Num points

20 30 40 50 60

— <http://vision.stanford.edu/teaching/cs231n-demos/knn/>

Nearest neighbor **HYPERPARAMETERS**

Hyperparameters – Siêu tham số



Hyperparameters – Siêu tham số

- What is the best value of k to use?
- What is the best distance to use?

- These are hyperparameters: choices about the algorithm that we set rather than learn.

- Very problem-dependent.
- Must try them all out and see what works best.

Setting Hyperparameters

- Idea #1: Choose hyperparameters that work best on the data

Your Dataset

- BAD.
- Hyperparameters:
 - + $k = 1$ always works perfectly on training data.
 - + *Distance metric*: no effect.

Setting Hyperparameters

- Idea #2: Split data into train and test, choose hyperparameters that work best on test data



- BAD.
- Hyperparameters:
 - + K : good on datatest.
 - + *Distance metric*: good on datatest.
- No idea how algorithm will perform on new data.

Setting Hyperparameters

- Idea #3: Split data into train, val, and test; choose hyperparameters on val and evaluate on test.



- Better!
- Hyperparameters:
 - + K : good on datavalidation.
 - + *Distance metric*: good on datavalidation.
- Have idea how algorithm will perform on new data (datatest).

Setting Hyperparameters

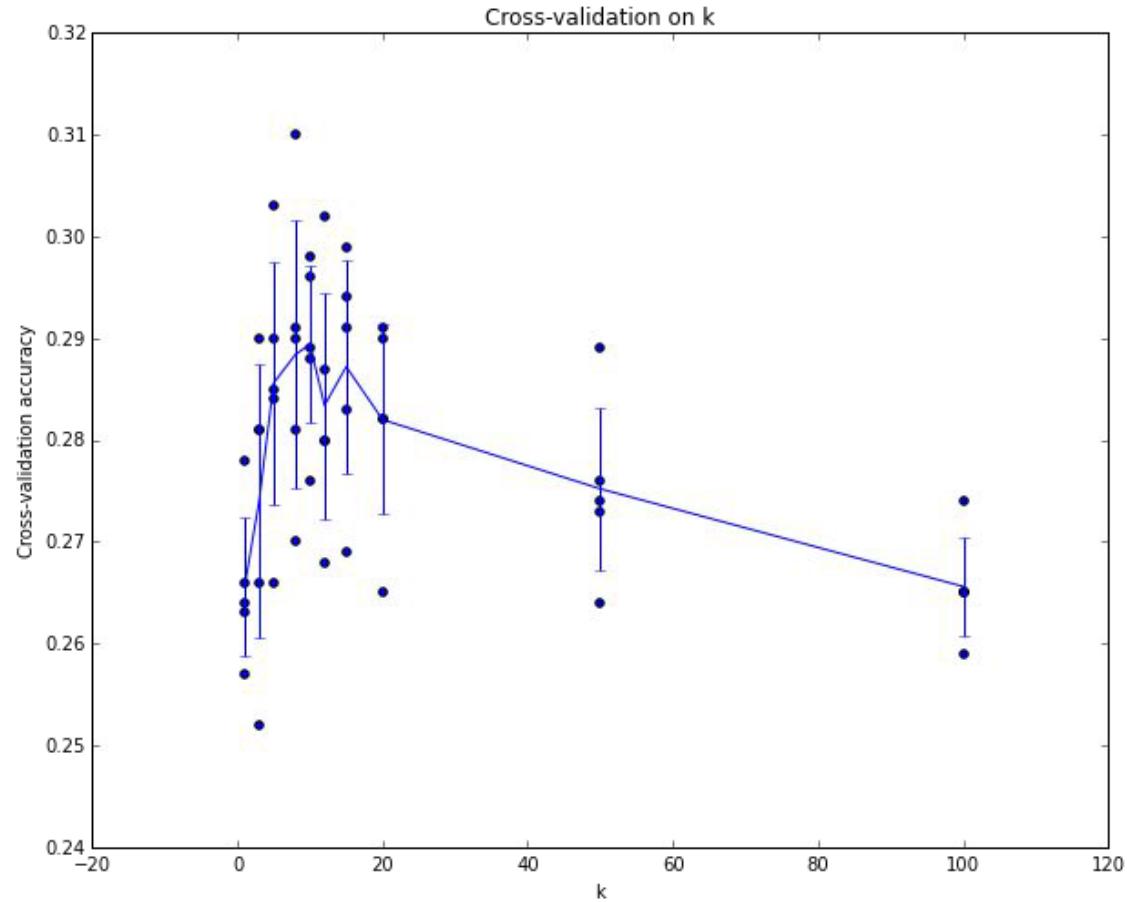
Your Dataset

- Idea #4: Cross-Validation: Split data into folds, try each fold as validation and average the results.

| | | | | | |
|--------|--------|--------|--------|--------|------|
| fold 1 | fold 2 | fold 3 | fold 4 | fold 5 | test |
| fold 1 | fold 2 | fold 3 | fold 4 | fold 5 | test |
| fold 1 | fold 2 | fold 3 | fold 4 | fold 5 | test |

- Useful for small datasets, but not used too frequently in deep learning.

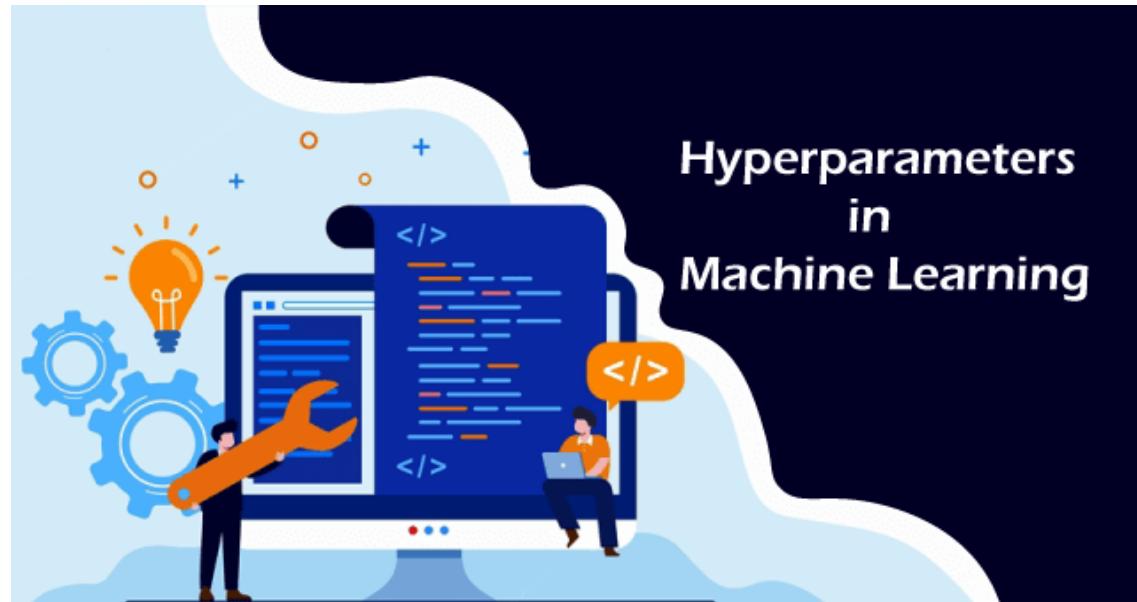
Setting Hyperparameters



- Example of 5-fold cross-validation for the value of k.
- Each point: single outcome.
- The line goes through the mean, bars indicated standard deviation.
- (Seems that $k \sim 7$ works best for this data)

Setting Hyperparameters

- Tập dữ liệu thu thập thông thường được chia làm mấy phần?



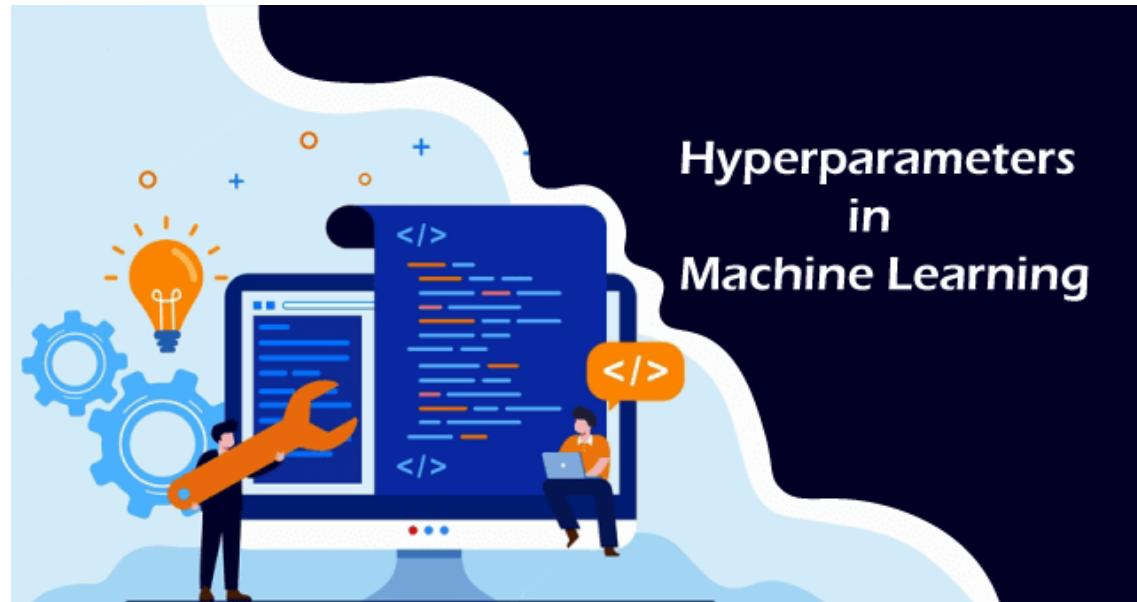
Setting Hyperparameters



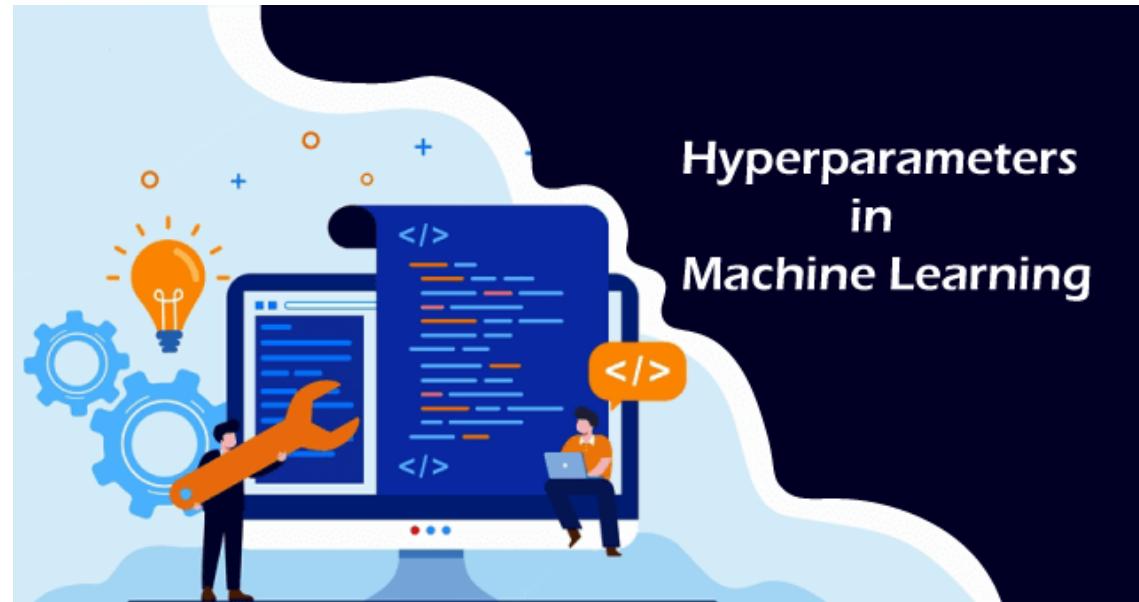
- Tập dữ liệu thu thập thông thường thường được chia làm mấy phần?
- Tập dữ liệu thu thập thông thường thường được chia làm ba phần.
 - + Train – Huấn luyện.
 - + Validation – Đánh giá.
 - + Test – Kiểm thử.

Setting Hyperparameters

- Thông thường khi huấn luyện mô hình người thường huấn luyện trên tập dữ liệu nào?



Setting Hyperparameters



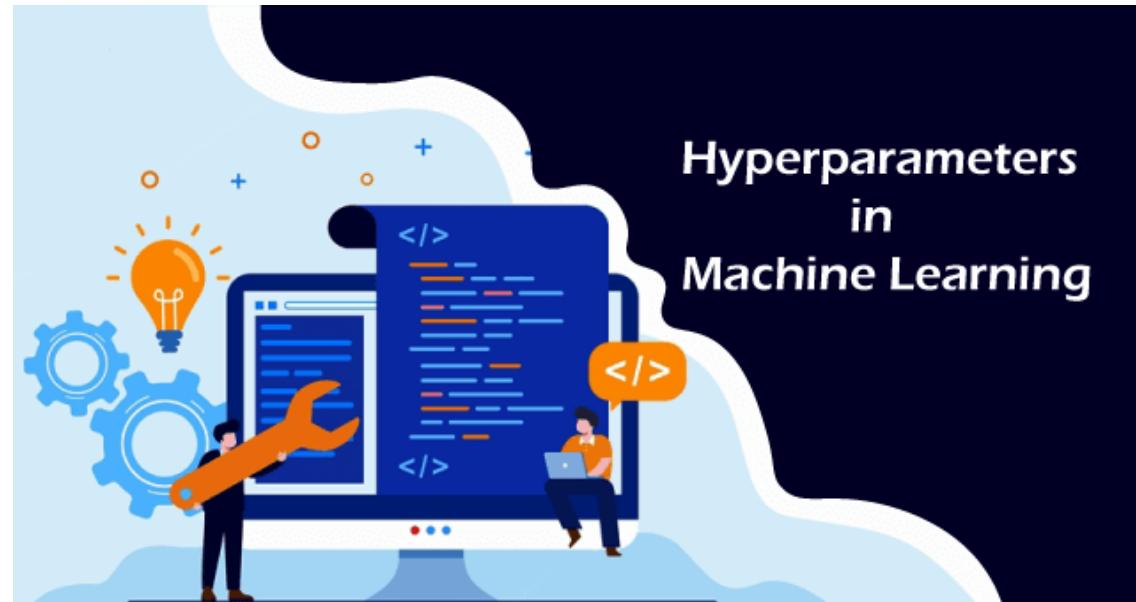
- Thông thường khi huấn luyện mô hình người thường huấn luyện trên tập dữ liệu nào?
- Thông thường khi huấn luyện mô hình người thường huấn luyện trên tập dữ liệu train.

Setting Hyperparameters



- Thông thường khi đánh giá khách quan mô hình người thường thực hiện trên tập dữ liệu nào?

Setting Hyperparameters



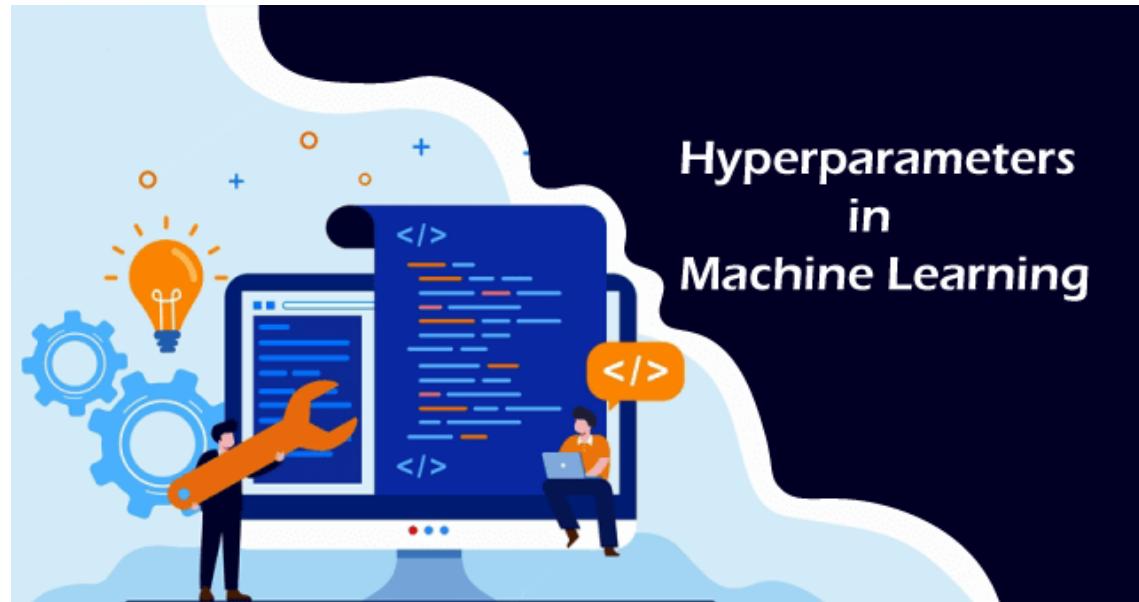
- Thông thường khi đánh giá khách quan mô hình người thường thực hiện trên tập dữ liệu nào?
- Thông thường khi đánh giá khách quan mô hình người thường thực hiện trên tập dữ liệu test.

Setting Hyperparameters



- Thông thường để tìm ra bộ siêu tham số thích hợp cho quá trình huấn luyện người ta thường thực hiện trên các bộ dữ liệu nào?

Setting Hyperparameters



- Thông thường để tìm ra bộ siêu tham số thích hợp cho quá trình huấn luyện người ta thường thực hiện trên các bộ dữ liệu nào?
- Thông thường để tìm ra bộ siêu tham số thích hợp cho quá trình huấn luyện người ta thường thực hiện trên bộ dữ liệu train và validation.

Nearest neighbor
DISCUSSION CONTINUE

K – NN on images never used.

- Very slow at test time
- Distance metrics on pixels are not informative

Original



Boxed



Shifted



Tinted



- All 3 images have same L2 distance to the one on the left

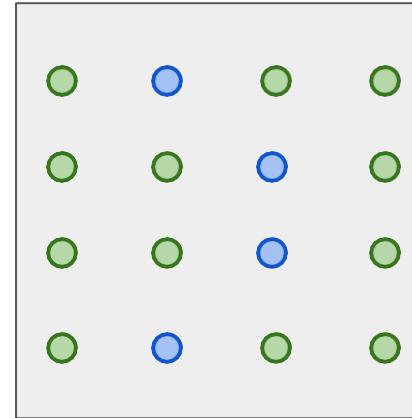
K – NN on images never used.

- Curse of dimensionality
- Lời nguyền chiều không gian

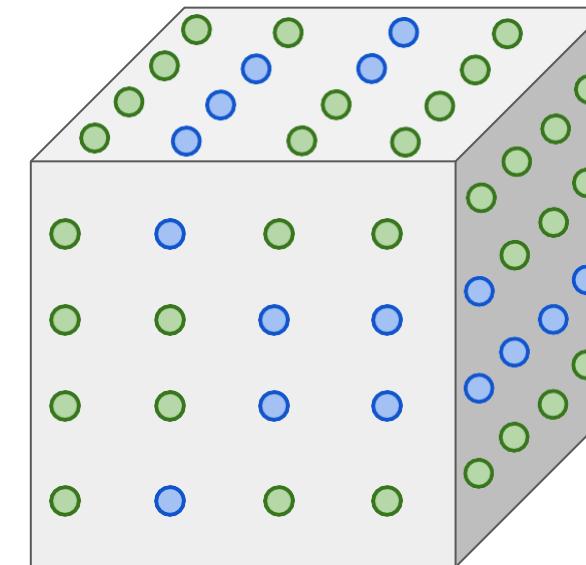
Dimensions = 1
Points = 4



Dimensions = 2
Points = 4^2



Dimensions = 3
Points = 4^3



Nearest neighbor **SUMMARY**

K-Nearest Neighbors: Summary

- In Image classification we start with a training set of images and labels, and must predict labels on the test set.
- Trong bài toán phân loại ảnh, chúng ta bắt đầu với tập huấn luyện gồm các ảnh và nhãn tương ứng, đồng thời phải dự đoán các nhãn trên tập dữ liệu kiểm thử.
- The K-Nearest Neighbors classifier predicts labels based on nearest training examples
- Thuật toán K – NN dự đoán nhãn dựa trên các mẫu (các điểm dữ liệu) huấn luyện gần nhất.

K-Nearest Neighbors: Summary

- Distance metric and K are hyperparameters.
- Độ đo khoảng cách và K là các siêu tham số trong quá trình học.
- Choose hyperparameters using the validation set; only run on the test set once at the very end!
- Chọn bộ siêu tham số bằng cách sử dụng bộ dữ liệu đánh giá; chỉ chạy trên tập kiểm thử một lần vào thời điểm cuối để đánh giá mô hình với bộ siêu tham số đã được lựa chọn.

CÂU HỎI ÔN TẬP

Câu hỏi

- Câu hỏi: Thuật toán Machine Learning được học trong bài giảng là thuật toán nào?

Câu hỏi

- Câu hỏi: Thuật toán Machine Learning được học trong bài giảng là thuật toán nào?
- Trả lời: Thuật toán Machine Learning được học trong bài giảng là thuật toán *k* Nearest Neighbor (*k*NN).

Câu hỏi

- Câu hỏi: Thuật toán *kNN* có thực hiện giai đoạn huấn luyện hay không?

Câu hỏi

- Câu hỏi: Thuật toán k NN có thực hiện giai đoạn huấn luyện hay không?
- Trả lời: Thuật toán k NN không thực hiện giai đoạn huấn luyện.

Câu hỏi

- Câu hỏi: Trong thuật toán *kNN*, giai đoạn huấn luyện được hiểu đơn giản là gì?

Câu hỏi

- Câu hỏi: Trong thuật toán *kNN*, giai đoạn huấn luyện được hiểu đơn giản là gì?
- Trả lời: Trong thuật toán *kNN*, giai đoạn huấn luyện được hiểu đơn giản là ghi nhớ toàn bộ tập dữ liệu huấn luyện.

Câu hỏi

- Câu hỏi: Trong thuật toán *kNN*, giai đoạn huấn luyện có độ phức tạp bao nhiêu?

Câu hỏi

- Câu hỏi: Trong thuật toán kNN , giai đoạn huấn luyện có độ phức tạp bao nhiêu?
- Trả lời: Trong thuật toán kNN , giai đoạn huấn luyện có độ phức tạp $O(1)$.

Câu hỏi

- Câu hỏi: Trong thuật toán *kNN*, giai đoạn kiểm thử có độ phức tạp bao nhiêu?

Câu hỏi

- Câu hỏi: Trong thuật toán kNN , giai đoạn kiểm thử có độ phức tạp bao nhiêu?
- Trả lời: Trong thuật toán kNN , giai đoạn kiểm thử có độ phức tạp $O(N)$ với N là số điểm dữ liệu trong bộ dữ liệu datatrain.

Câu hỏi

- Câu hỏi: Trong thuật toán *kNN* nhãn của dữ liệu kiểm tra được dự báo như thế nào?

Câu hỏi

- Câu hỏi: Trong thuật toán *kNN* nhãn của dữ liệu kiểm tra được dự báo như thế nào?
- Trả lời: Trong thuật toán *kNN* nhãn của dữ liệu kiểm tra được dự báo bằng nhãn của điểm dữ liệu huấn luyện gần nhất ($k = 1$).

Câu hỏi

- Câu hỏi: Trong thuật toán *kNN* nhãn của dữ liệu kiểm tra được dự báo như thế nào?
- Trả lời: Trong thuật toán *kNN* nhãn của dữ liệu kiểm tra được dự báo bằng nhãn của điểm dữ liệu huấn luyện gần nhất ($k = 1$).
- Trả lời: Trong thuật toán *kNN* nhãn của dữ liệu kiểm tra bằng cách xem xét k điểm dữ liệu gần nhất trong tập huấn luyện. Nhãn của điểm mới được quyết định dựa trên số đồng nhãn của k hàng xóm gần nhất (k tổng quát).

Câu hỏi

— Câu hỏi: Các bước chính trong thuật toán *kNN* là gì?

Câu hỏi

- Câu hỏi: Các bước chính trong thuật toán *kNN* là gì?
- Trả lời: Các bước chính trong thuật toán *kNN* là:
 - + Chọn số lượng hàng xóm *k*.
 - + Tính toán khoảng cách giữa điểm dữ liệu mới và tất cả các điểm dữ liệu trong tập huấn luyện.
 - + Sắp xếp các khoảng cách và chọn *k* điểm gần nhất.
 - + Xác định nhãn dựa trên số đông của *k* hàng xóm gần nhất.

Câu hỏi

- Câu hỏi: Trong thuật toán *kNN* các siêu tham số (hyperparameter) là gì?

Câu hỏi

- Câu hỏi: Trong thuật toán *kNN* các siêu tham số (hyperparameter) là gì?
- Trả lời: Trong thuật toán *kNN* các siêu tham số (hyperparameter) bao gồm *k* và độ đo khoảng cách.

Câu hỏi

- Câu hỏi: Trong bài giảng trình bày mấy độ đo khoảng cách? Hãy kể tên các độ đo khoảng cách ấy?

Câu hỏi

- Câu hỏi: Trong bài giảng trình bày mấy độ đo khoảng cách? Hãy kể tên các độ đo khoảng cách ấy?
- Trả lời: Trong bài giảng trình bày 2 độ đo khoảng cách. Tên các độ đo khoảng cách là:
 - + Manhattan.
 - + Euclid.

Câu hỏi

— Câu hỏi: Làm thế nào để chọn giá trị của k trong k NN?

Câu hỏi

- Câu hỏi: Làm thế nào để chọn giá trị của k trong kNN ?
- Trả lời: Giá trị của k thường được chọn thông qua thực nghiệm và kiểm tra chéo. Giá trị nhỏ của k có thể dẫn đến mô hình bị quá khớp, trong khi giá trị lớn của k có thể làm mô hình bị khớp thiếu vì mô hình trở nên quá tổng quát và bỏ qua các đặc trưng quan trọng.

Câu hỏi

- Câu hỏi: Bạn có thể giải thích sự khác biệt giữa khoảng cách Euclid và khoảng cách Manhattan?

Câu hỏi

- Câu hỏi: Bạn có thể giải thích sự khác biệt giữa khoảng cách Euclid và khoảng cách Manhattan?
- Trả lời:
 - + Khoảng cách Euclid: Tính bằng cách lấy căn bậc hai của tổng bình phương các khác biệt giữa các chiều tọa độ tương ứng.
 - + Khoảng cách Manhattan: Tính bằng tổng các giá trị tuyệt đối của các khác biệt giữa các chiều tọa độ tương ứng.

BÀI TẬP KNN

Bài tập *kNN*

- Bài tập 11:
 - + Load bộ dữ liệu uitTinyTrainCIFAR10 trong thư mục làm việc của Colab.
 - + Xây dựng mô hình bằng thuật toán *kNN* với $k = 1$ và sử dụng độ đo khoảng cách Manhattan.
 - + Tính độ chính xác của mô hình kết quả trên bộ dữ liệu uitTinyTrainCIFAR10 , uitTinyValidationCIFAR10 và uitTinyTestCIFAR10.

Bài tập *kNN*

- Bài tập 12:
 - + Load bộ dữ liệu uitTinyTrainCIFAR10 trong thư mục làm việc của Colab.
 - + Xây dựng mô hình bằng thuật toán *kNN* với $k = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20$ và sử dụng độ đo khoảng cách Manhattan.
 - + Chỉ ra siêu tham số *k* tốt nhất.

Bài tập *k*NN

- Bài tập 13:
 - + Load bộ dữ liệu uitTinyTrainCIFAR10 trong thư mục làm việc của Colab.
 - + Xây dựng mô hình bằng thuật toán *k*NN với $k = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20$ và sử dụng độ đo khoảng cách Euclidean.
 - + Chỉ ra siêu tham số *k* tốt nhất.

Bài tập *kNN*

— Bài tập 14:

- + Load bộ dữ liệu uitTinyTrainCIFAR10 trong thư mục làm việc của Colab.
- + Xây dựng mô hình bằng thuật toán *kNN* với $k = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20$ và sử dụng độ đo khoảng cách Euclidean, Manhattan.
- + Chỉ ra siêu tham số độ đo khoảng cách và k tốt nhất.

Bài tập *k*NN

— Bài tập 15:

- + Load bộ dữ liệu uitSmallTrainCIFAR10 trong thư mục làm việc của Colab.
- + Xây dựng mô hình bằng thuật toán *k*NN với $k = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20$ và sử dụng độ đo khoảng cách Euclidean, Manhattan.
- + Chỉ ra siêu tham số độ đo khoảng cách và k tốt nhất.

Bài tập *kNN*

— Bài tập 16:

- + Load bộ dữ liệu uitMediumTrainCIFAR10 trong thư mục làm việc của Colab.
- + Xây dựng mô hình bằng thuật toán *kNN* với $k = 1, 3, 5, 7, 9$ và sử dụng độ đo khoảng cách Euclidean, Manhattan.
- + Chỉ ra siêu tham số độ đo khoảng cách và k tốt nhất.

LINEAR CLASSIFICATION

Linear classification

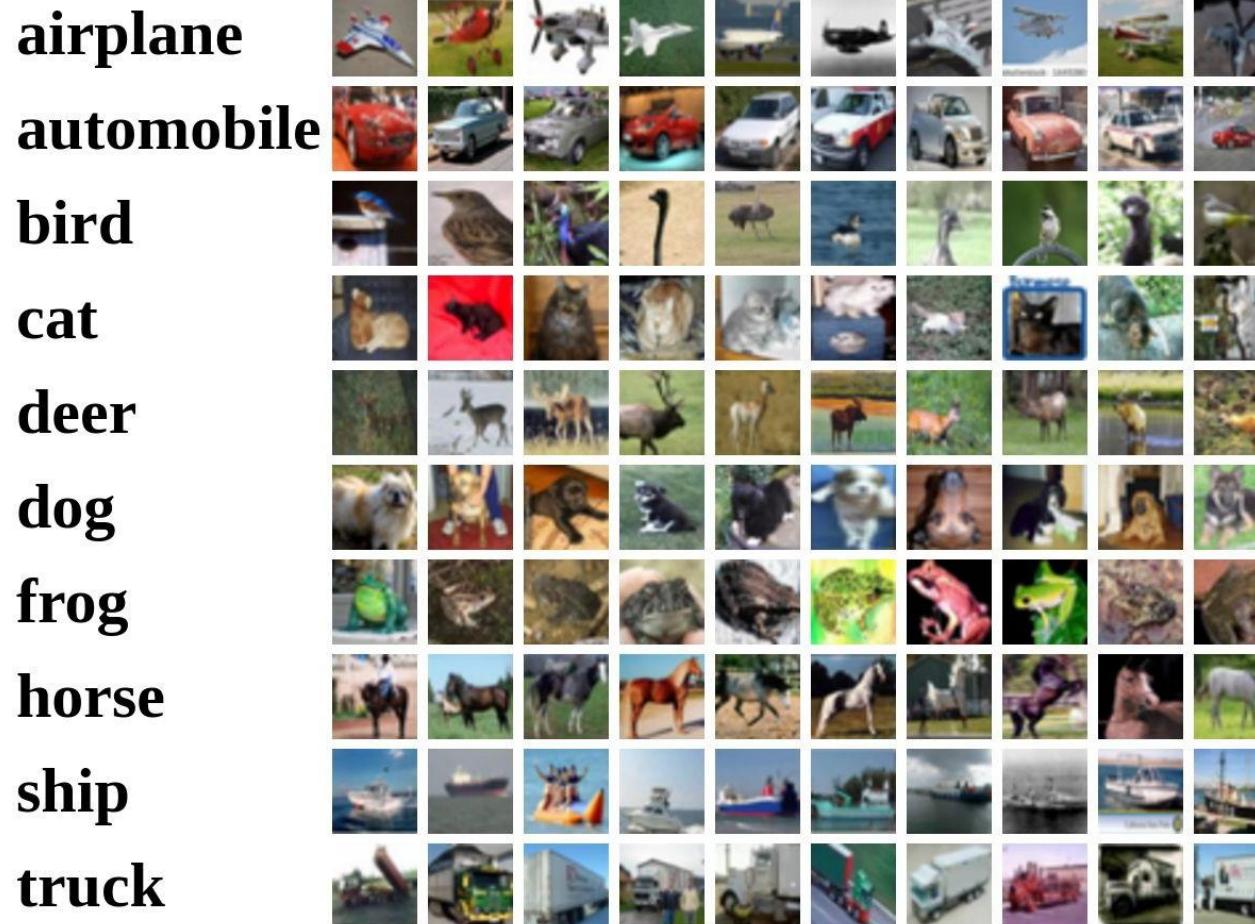


Neural Network

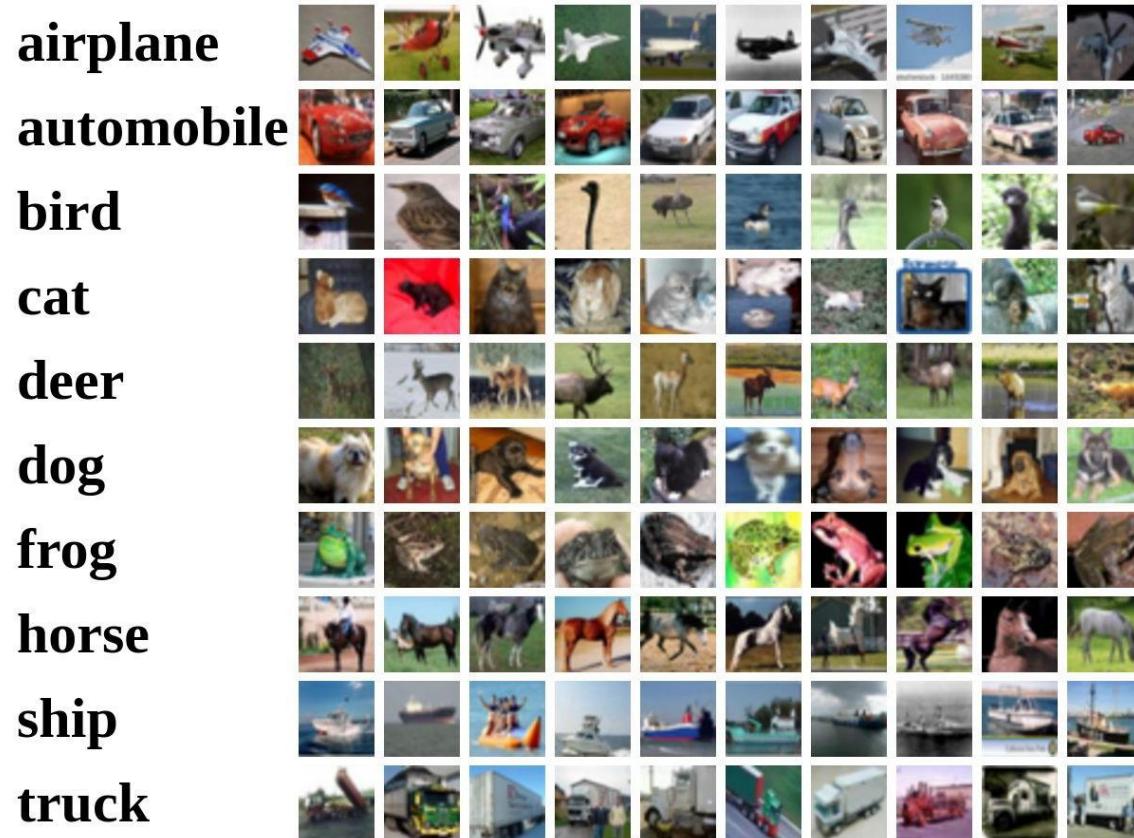
Linear classifiers

Linear classifiers

Dataset: CIFAR10



Dataset: CIFAR10



— Lịch sử:

- + Công bố 2009.
- + Bộ dữ liệu Cifar10 được phát triển bởi nhóm nghiên cứu tại Đại học Toronto.
- + Nhóm tác giả: Alex Krizhevsky, Vinod Nair và Geoffrey Hinton.

Dataset: CIFAR10

airplane



automobile



bird



cat



deer



dog



frog



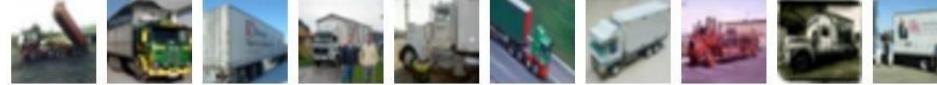
horse



ship



truck



— 10 classes.

*airplane
automobile
bird
cat
deer
dog
frog
horse
ship
truck*

Dataset: CIFAR10

airplane



automobile



bird



cat



deer



dog



frog



horse



ship



truck

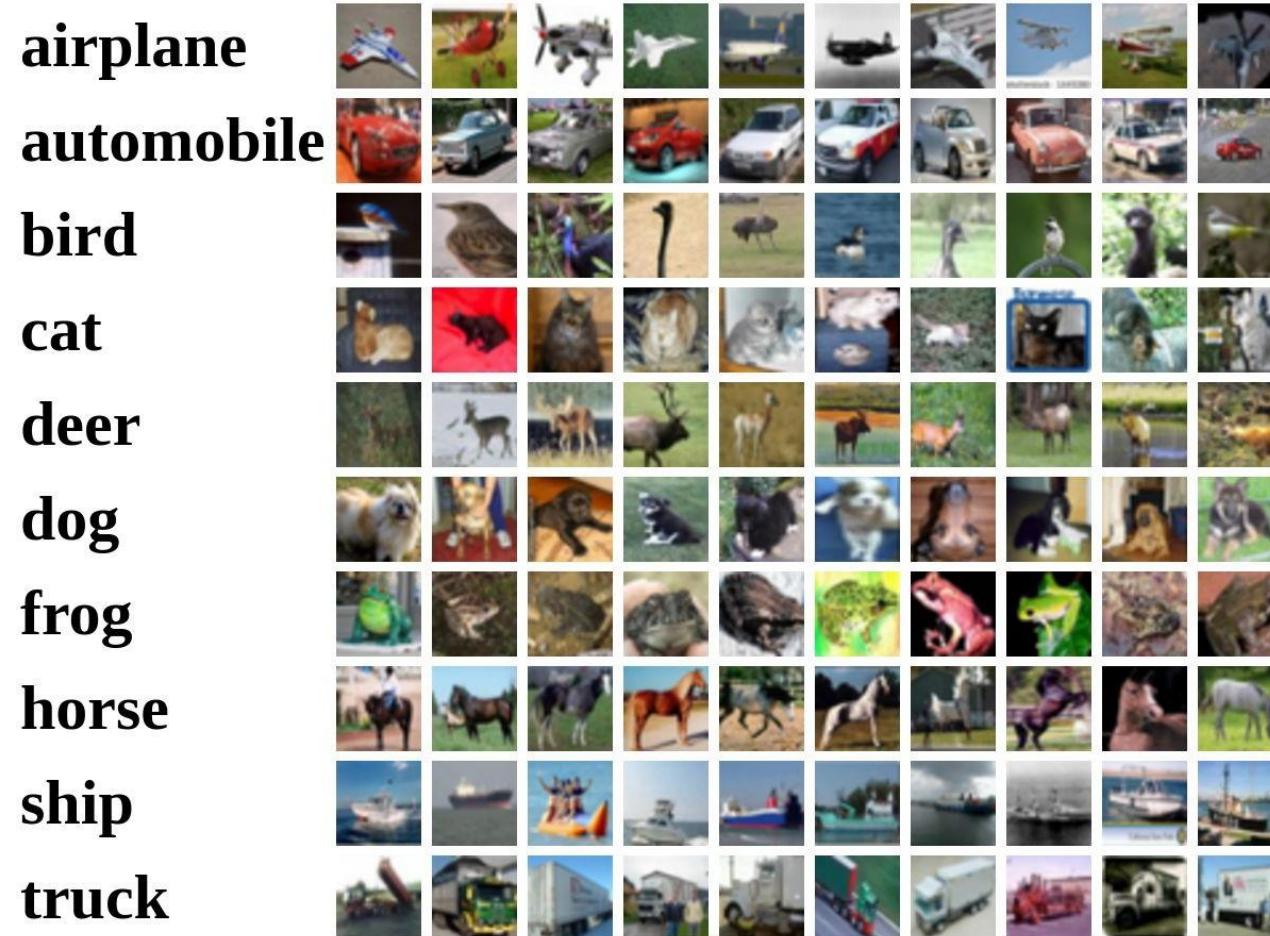


– 10 classes.

*airplane
automobile
bird
cat
deer
dog
frog
horse
ship
truck*

$$= \begin{pmatrix} 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \end{pmatrix}$$

Dataset: CIFAR10



– Datatrain of CIFAR10:

$$\mathcal{D}_{train} = \{(x^{(i)}, y^{(i)})\}_{i=0}^{N-1}$$

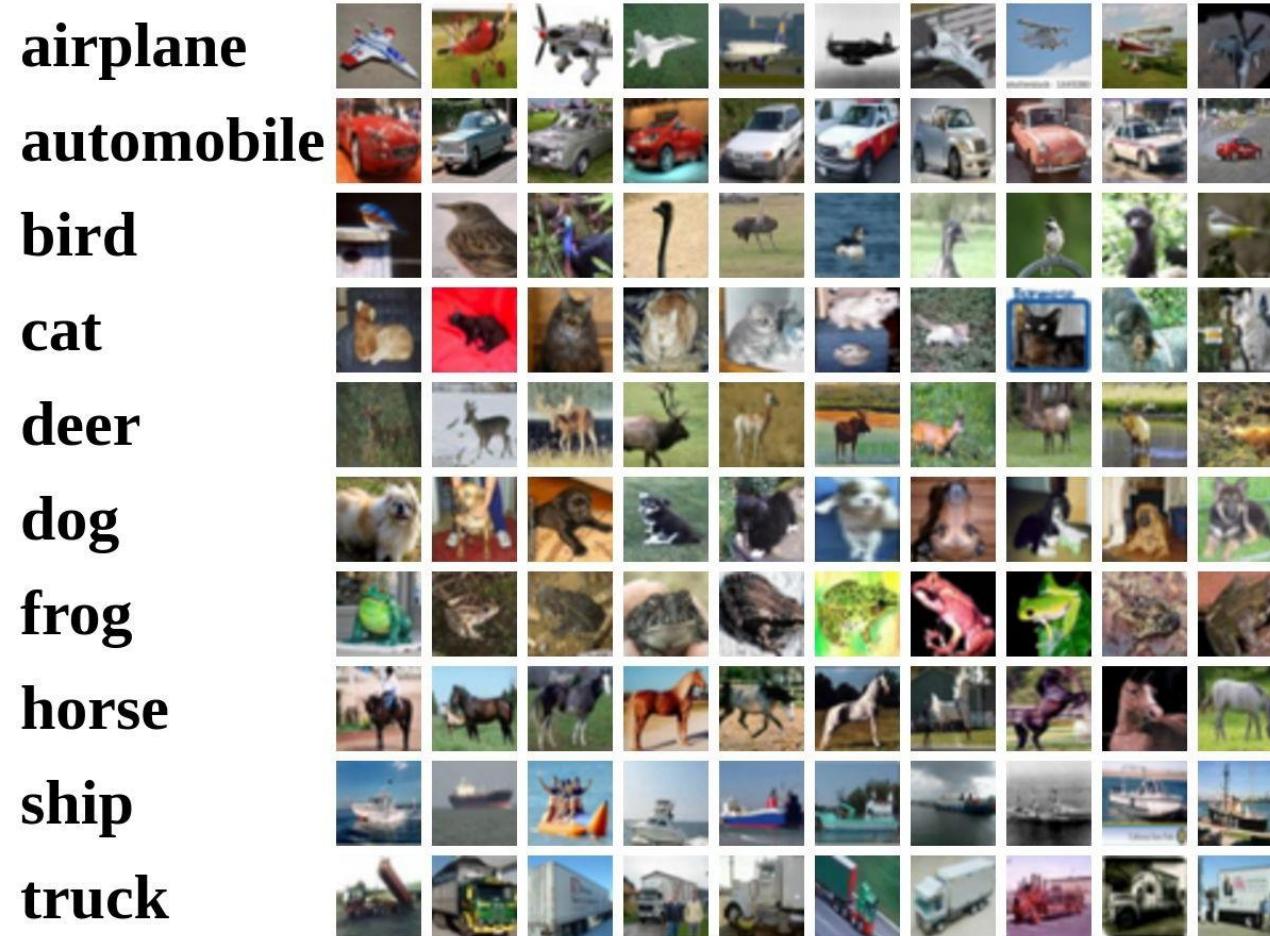
– Where:

+ N : 50.000.

+ $x^{(i)}$ ảnh train thứ i có kích thước $32 \times 32 \times 3$.

+ $y^{(i)}$ nhãn ảnh thứ $i \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$.

Dataset: CIFAR10



– Datatest of CIFAR10:

$$\mathcal{D}_{test} = \{(x^{(i)}, y^{(i)})\}_{i=0}^{M-1}$$

– Where:

+ M : 10.000.

+ $x^{(i)}$ ảnh test thứ i có kích thước $32 \times 32 \times 3$.

+ $y^{(i)}$ nhãn ảnh thứ $i \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$.

Parametric Approach: Linear Classifier

Image



Parametric Approach: Linear Classifier

Image



airplane



automobile



bird



cat



deer



dog



frog



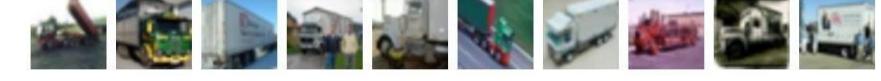
horse



ship



truck



Parametric Approach: Linear Classifier

Image



10 numbers giving
class scores

Parametric Approach: Linear Classifier

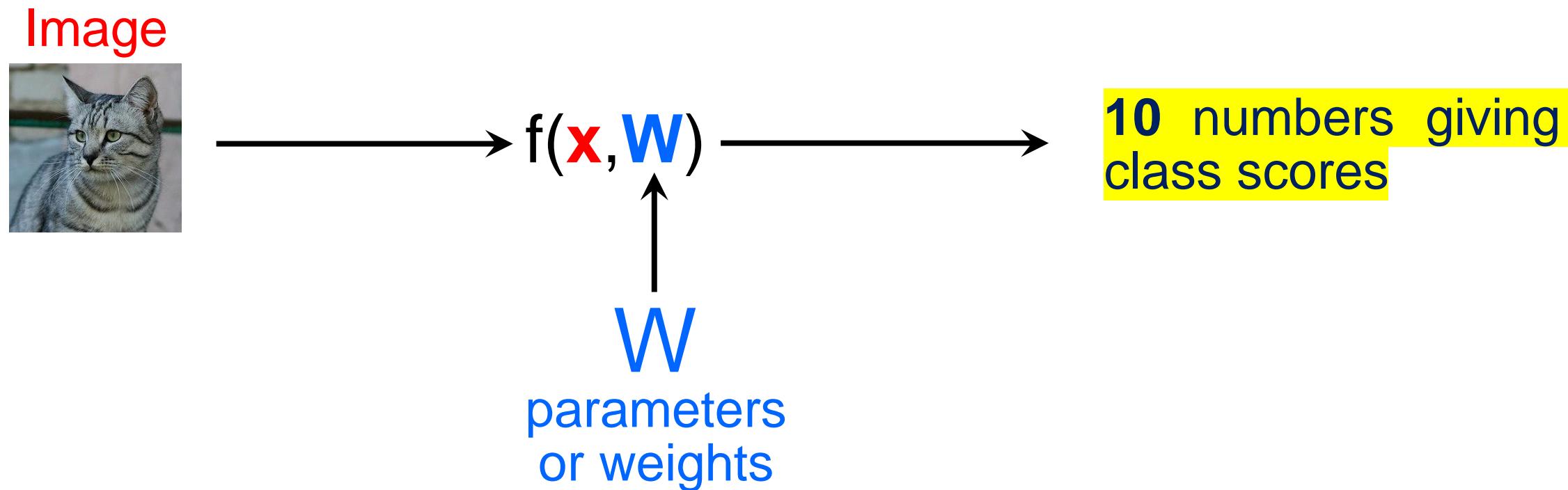
Image



$$\longrightarrow f(\mathbf{x}, \mathbf{W}) \longrightarrow$$

10 numbers giving
class scores

Parametric Approach: Linear Classifier

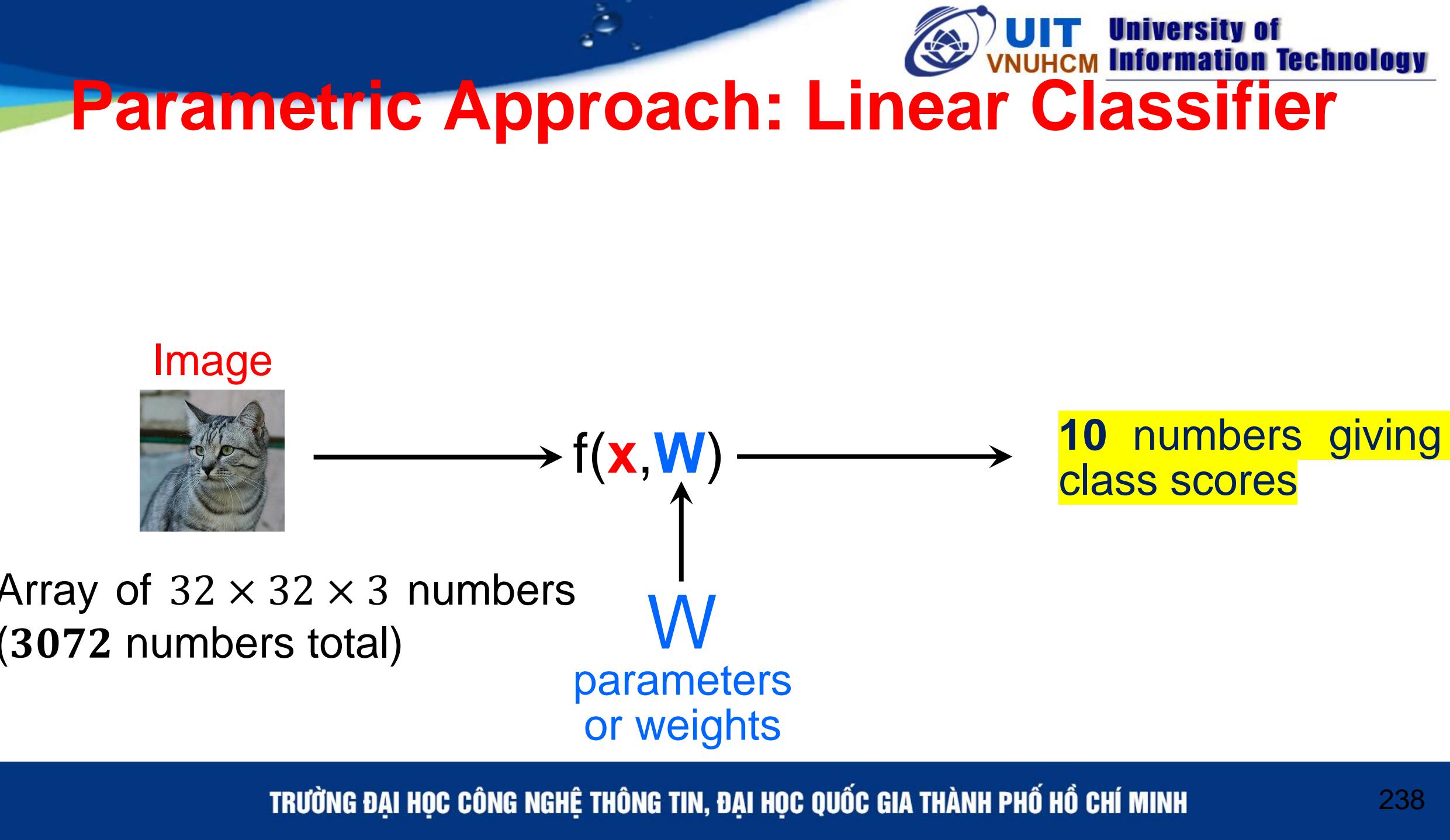


Parametric Approach: Linear Classifier



Image

Array of $32 \times 32 \times 3$ numbers
(3072 numbers total)



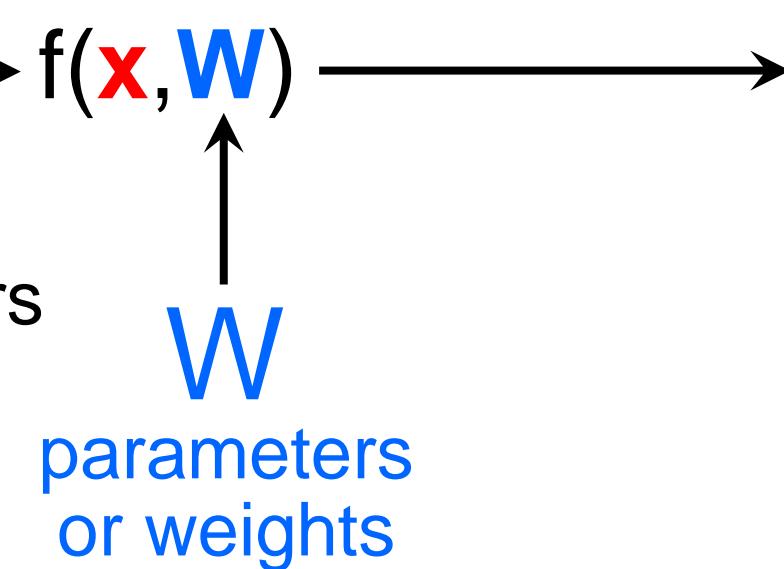
Parametric Approach: Linear Classifier

$$f(\mathbf{x}, \mathbf{W}) = \mathbf{W} \cdot \mathbf{x}$$

Image



Array of $32 \times 32 \times 3$ numbers
(3072 numbers total)



10 numbers giving
class scores

Parametric Approach: Linear Classifier

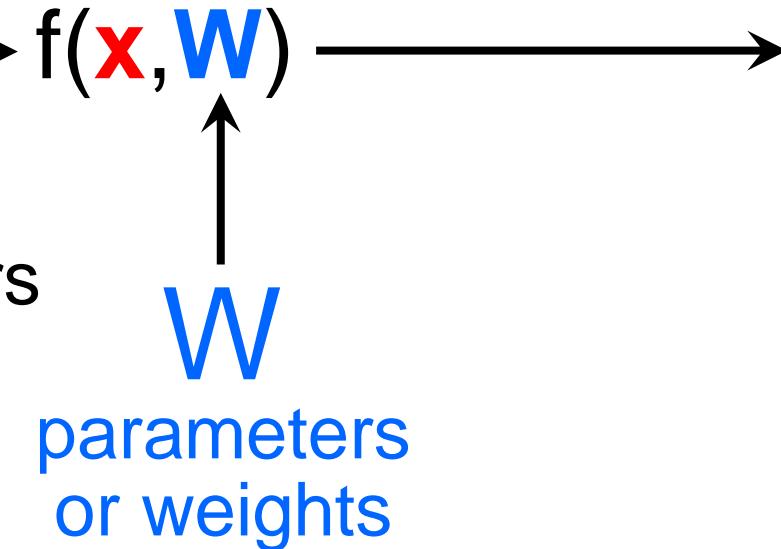
10×1

$$f(\mathbf{x}, \mathbf{W}) = \mathbf{W} \mathbf{x}$$

Image

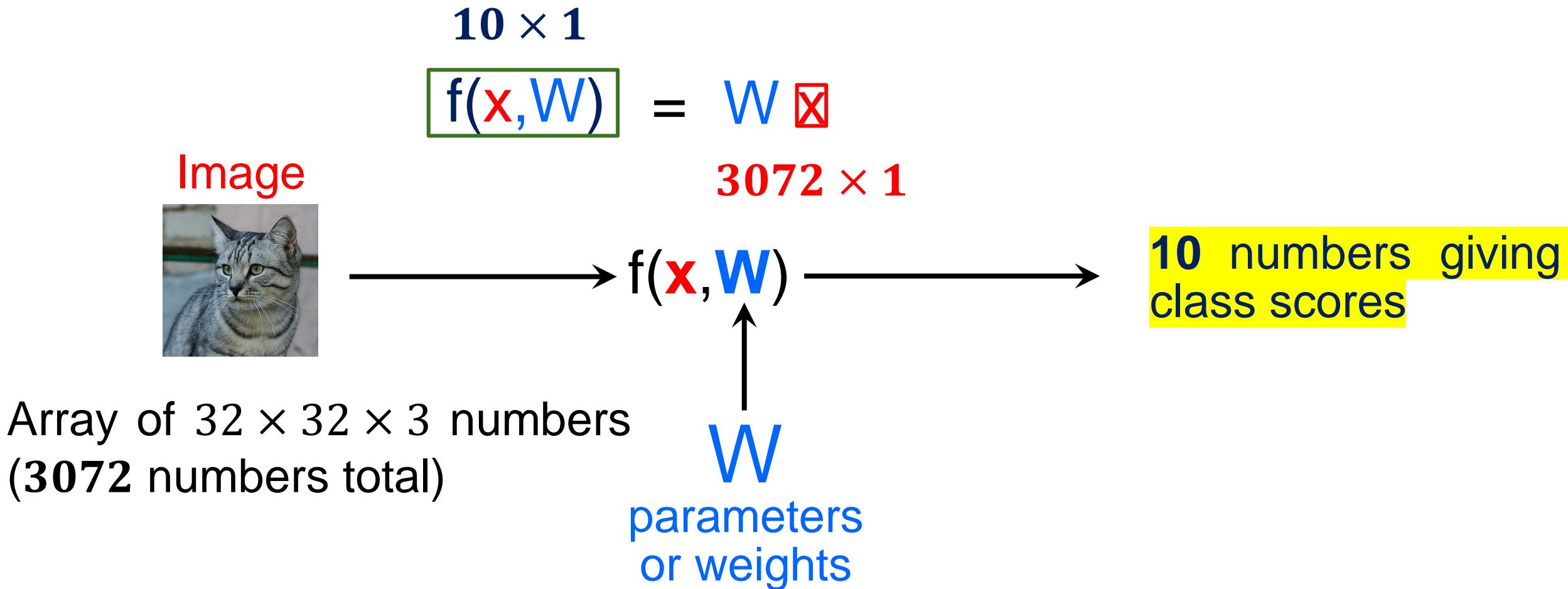


Array of $32 \times 32 \times 3$ numbers
(3072 numbers total)

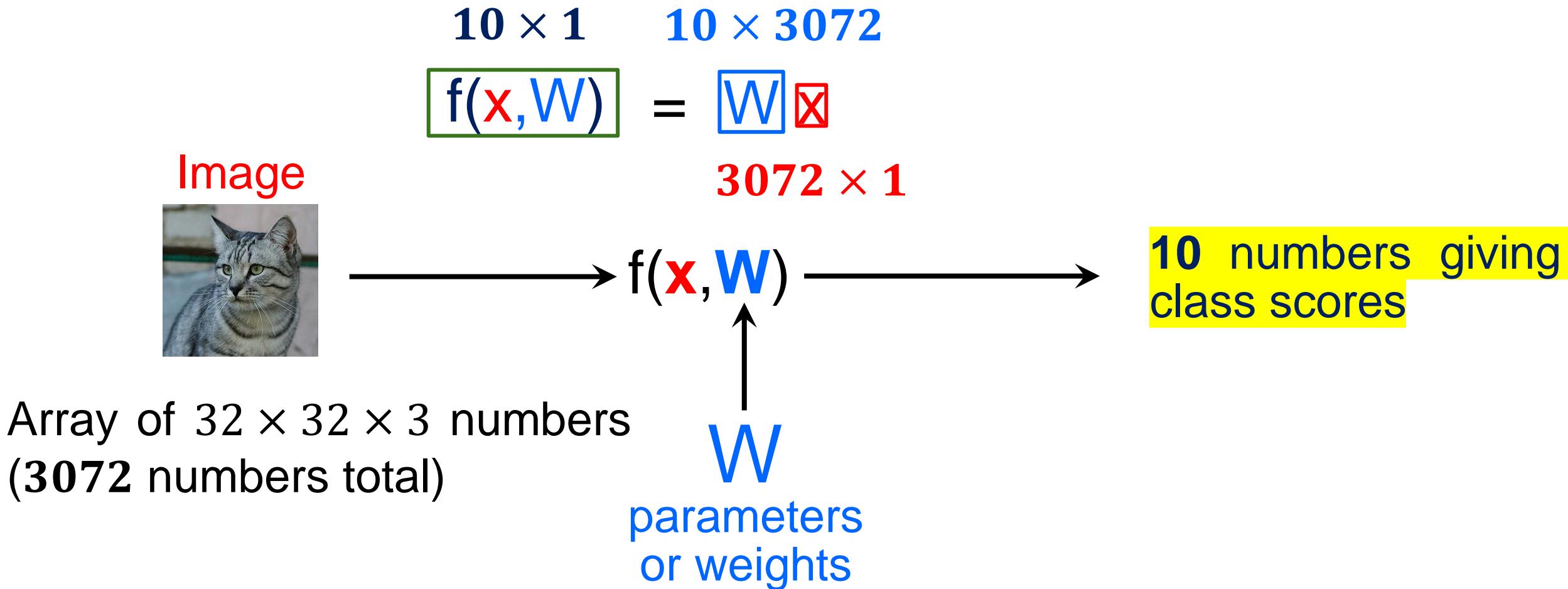


10 numbers giving
class scores

Parametric Approach: Linear Classifier



Parametric Approach: Linear Classifier



Parametric Approach: Linear Classifier

$$f(x, W) = W \otimes x + b$$

Image



Array of $32 \times 32 \times 3$ numbers
(3072 numbers total)

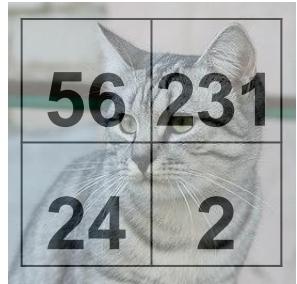
$$\rightarrow f(x, W) \rightarrow$$

3072×1

W
parameters
or weights

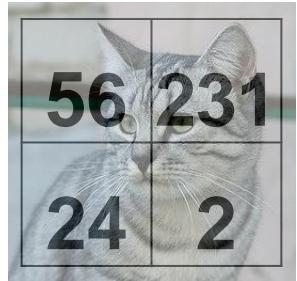
10 numbers giving
class scores

Ex: image 1 channel, 4 pixel



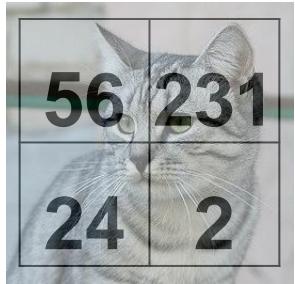
Input
image

Ex: image 1 channel, 4 pixel, 3 classes



Input
image

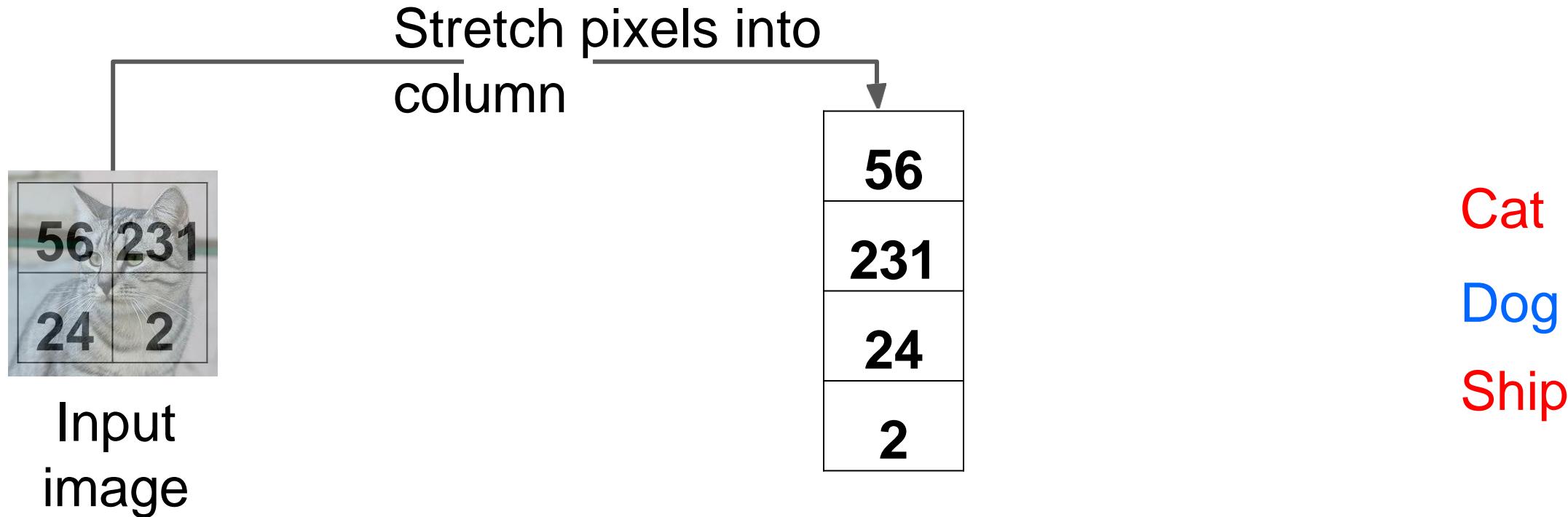
Ex: image 1 channel, 4 pixel, 3 classes



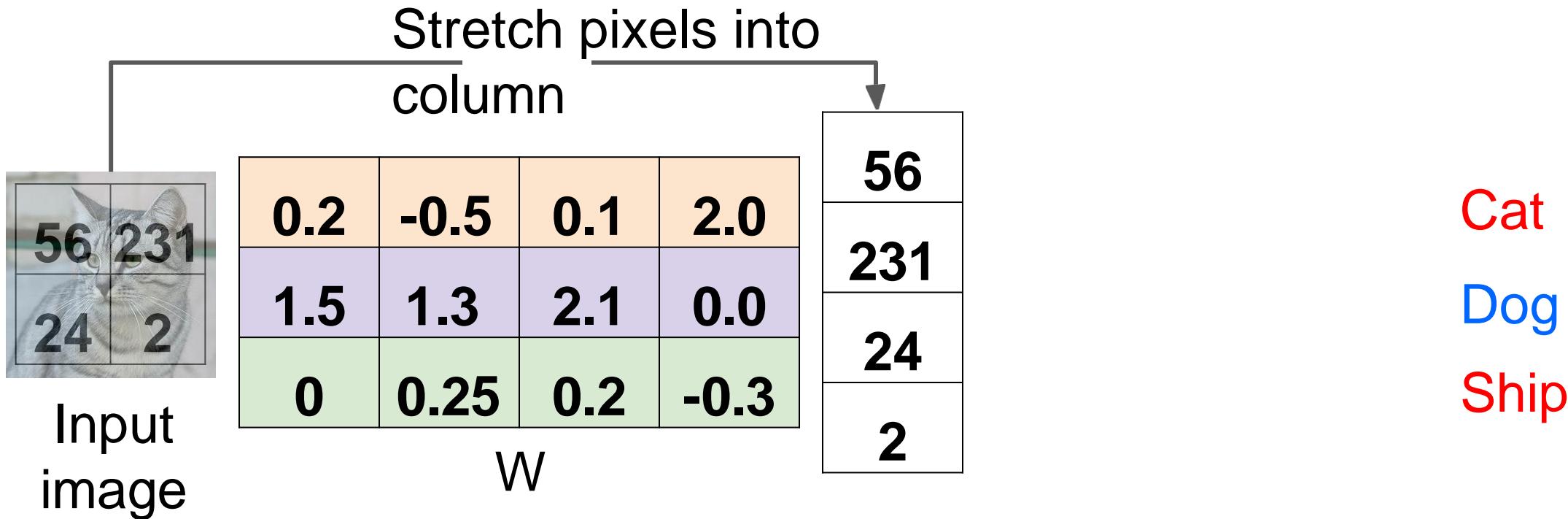
Input
image

Cat
Dog
Ship

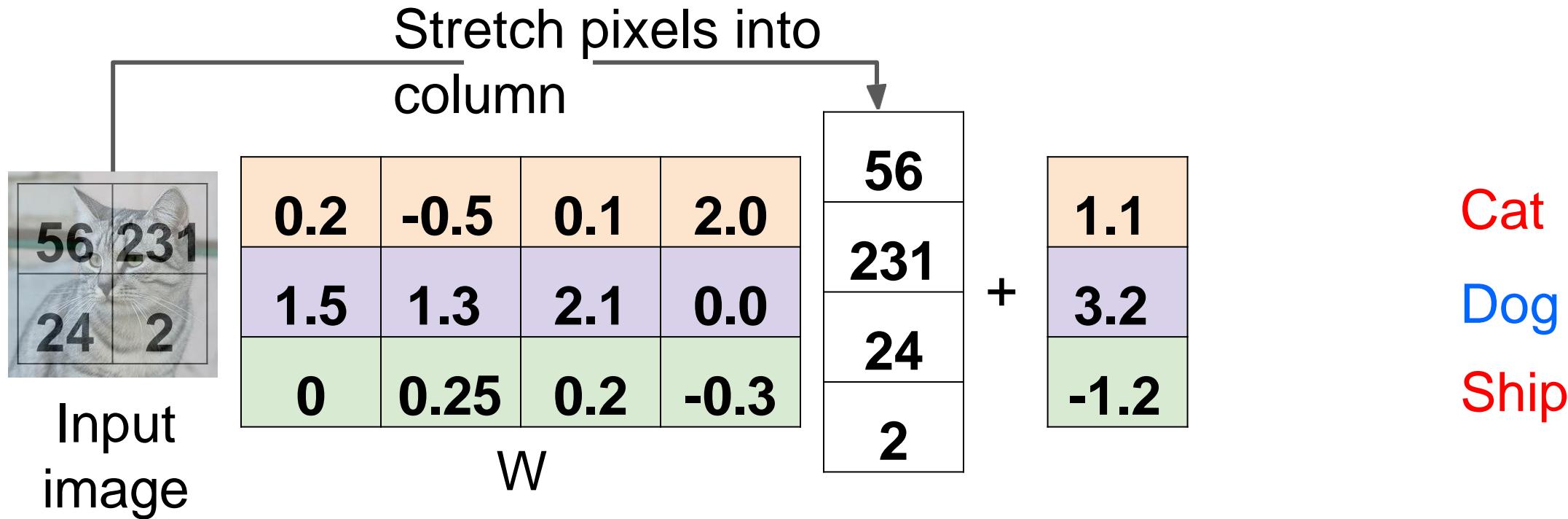
Ex: image 1 channel, 4 pixel, 3 classes



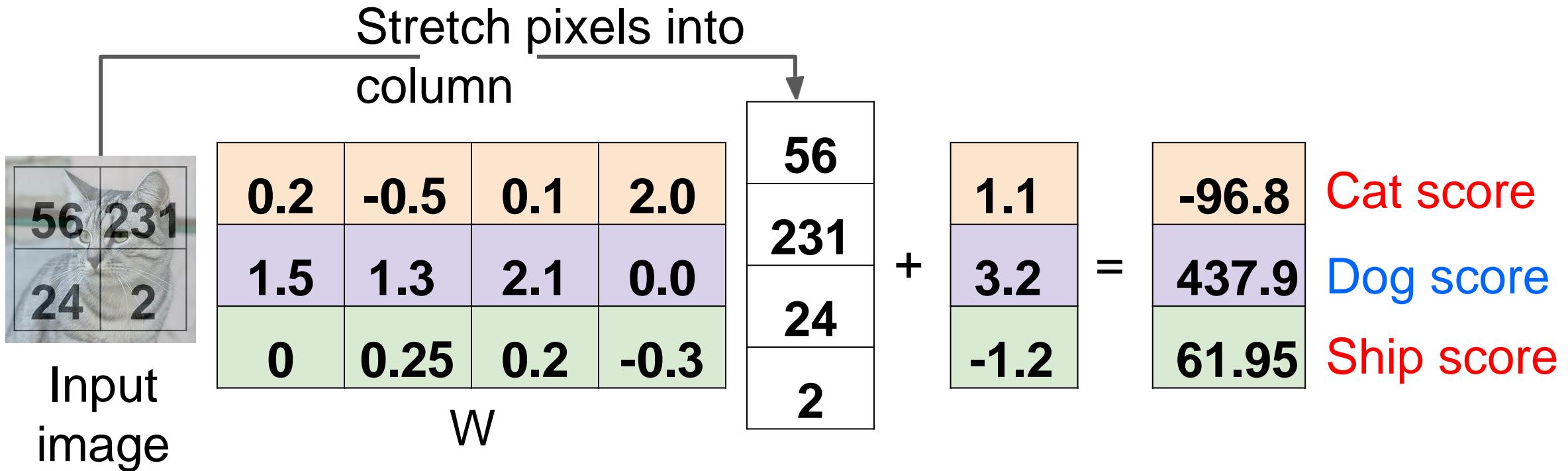
Ex: image 1 channel, 4 pixel, 3 classes



Ex: image 1 channel, 4 pixel, 3 classes



Ex: image 1 channel, 4 pixel, 3 classes



Parametric Approach: Linear Classifier

- Mô hình Linear Classifier:

$$f(x, W) = W \cdot x + b = score$$

- Trong đó:

- + x : ảnh.

- + W : ma trận trọng số.

- + b : hệ số *bias*.

- + W, b : model, classifier.

airplane



automobile



bird



cat



deer



dog



frog



horse



ship



truck



Parametric Approach: Linear Classifier

— Mô hình Linear Classifier:

$$f(x, W) = W \cdot x + b = \boxed{\text{score}}$$

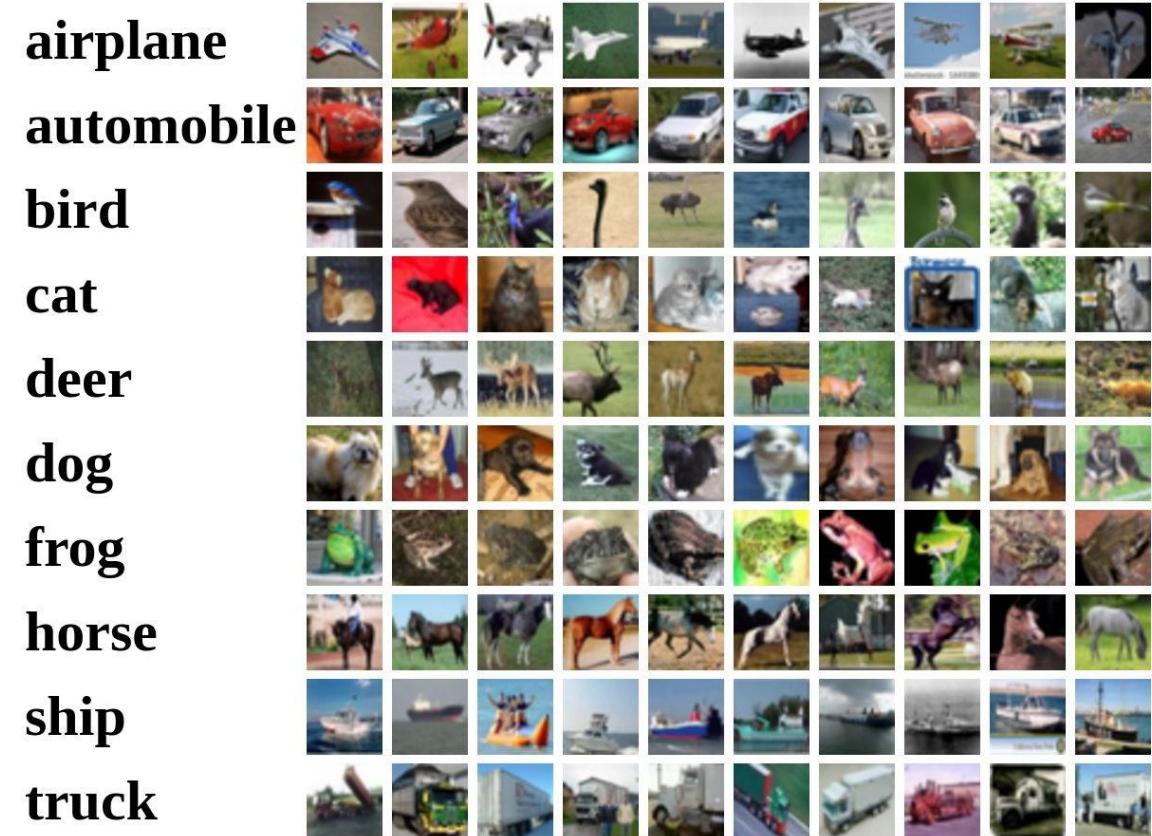
— Trong đó:

+ x : ảnh.

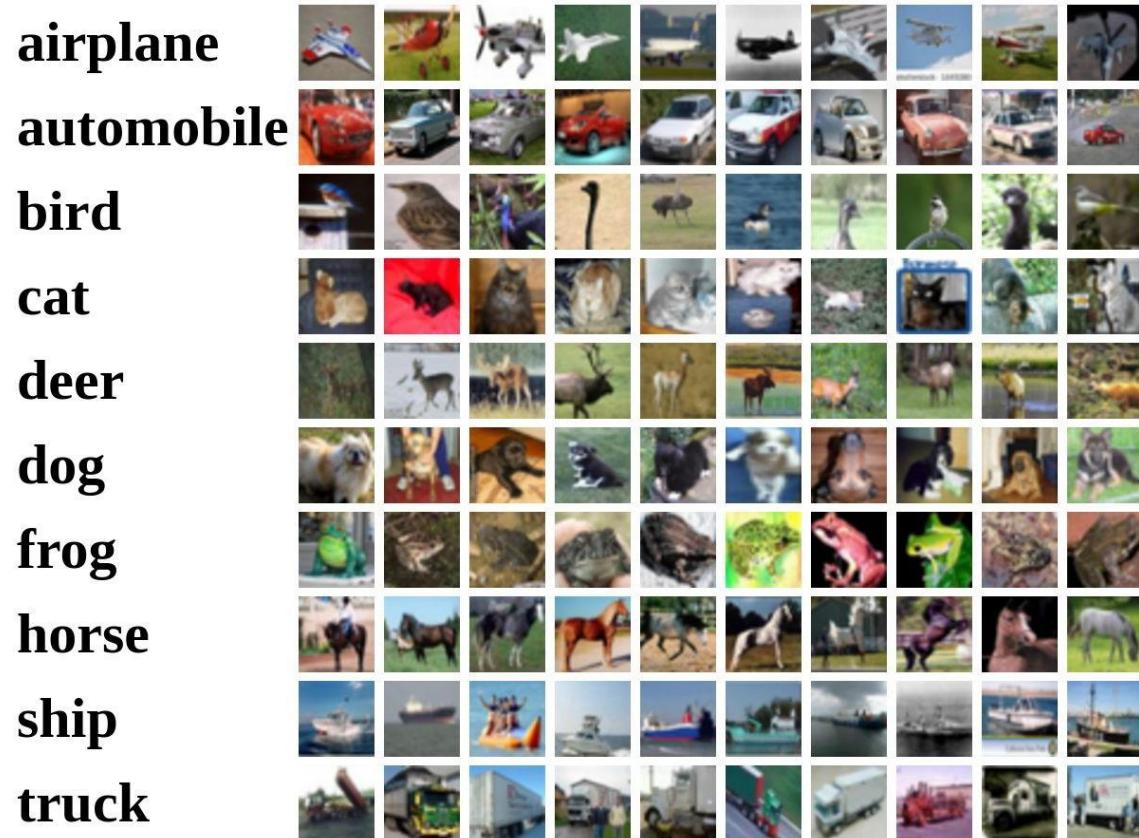
+ W : ma trận trọng số.

+ b : hệ số *bias*.

+ W, b : model, classifier.



Parametric Approach: Linear Classifier



— Véc tơ *score*:

$$score = \begin{pmatrix} s_0 \\ s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ s_6 \\ s_7 \\ s_8 \\ s_9 \end{pmatrix} \equiv \begin{pmatrix} airplane \\ automobile \\ bird \\ cat \\ deer \\ dog \\ frog \\ horse \\ ship \\ truck \end{pmatrix}$$

Parametric Approach: Linear Classifier

— Mô hình Linear Classifier:

$$f(x, W) = W \cdot x + b = score$$

— Trong đó:

+ x : ảnh.

+ W : ma trận trọng số.

+ b : hệ số *bias*.

+ W, b : model, classifier.

airplane



automobile



bird



cat



deer



dog



frog



horse



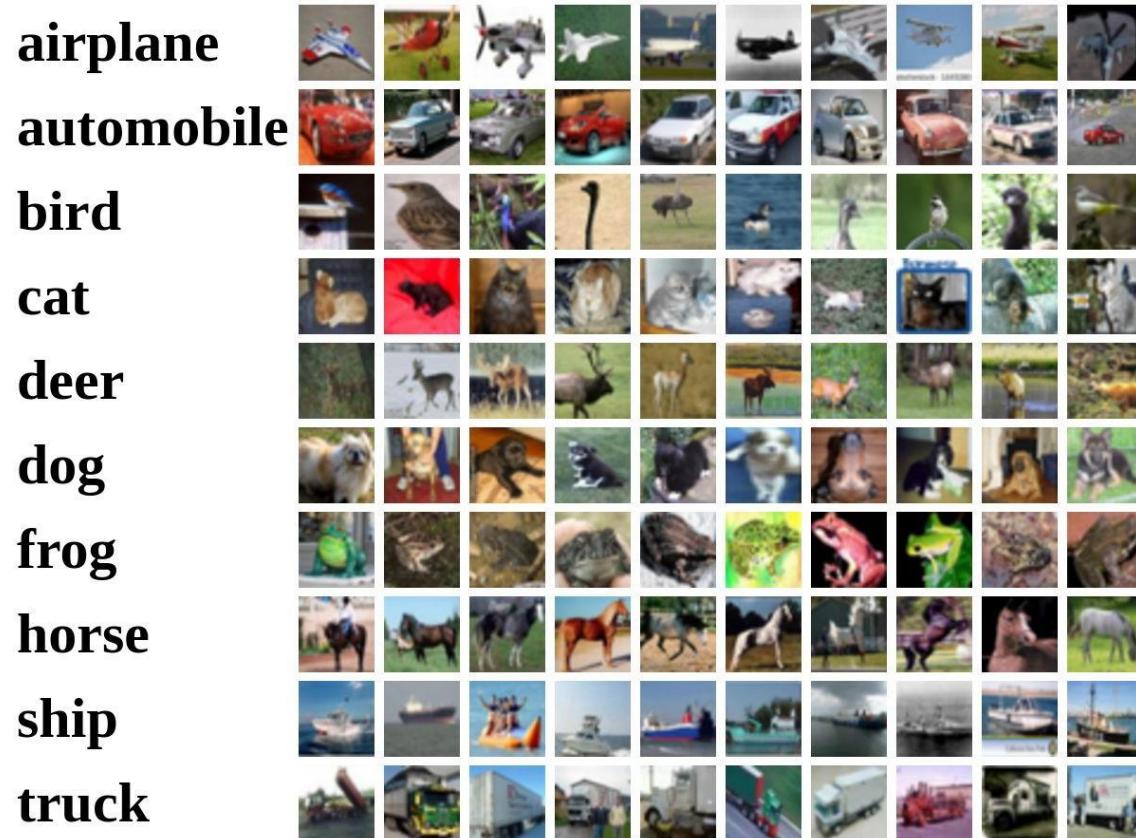
ship



truck



Parametric Approach: Linear Classifier



– Hệ số *bias*:

$$b = \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \\ b_8 \\ b_9 \end{pmatrix}$$

Parametric Approach: Linear Classifier

— Mô hình Linear Classifier:

$$f(x, W) = W \boxed{x} + b = score$$

— Trong đó:

+ x : ảnh.

+ W : ma trận trọng số.

+ b : hệ số *bias*.

+ W, b : model, classifier.

airplane



automobile



bird



cat



deer



dog



frog



horse



ship



truck



Parametric Approach: Linear Classifier

airplane



automobile



bird



cat



deer



dog



frog



horse



ship



truck



- Training image:
- Size: $32 \times 32 \times 3$.

$$x = \begin{pmatrix} x_0 \\ x_1 \\ \vdots \\ x_i \\ \vdots \\ x_{3069} \\ x_{3070} \\ x_{3071} \end{pmatrix}$$

Parametric Approach: Linear Classifier

— Mô hình Linear Classifier:

$$f(x, W) = \boxed{W} x + b = score$$

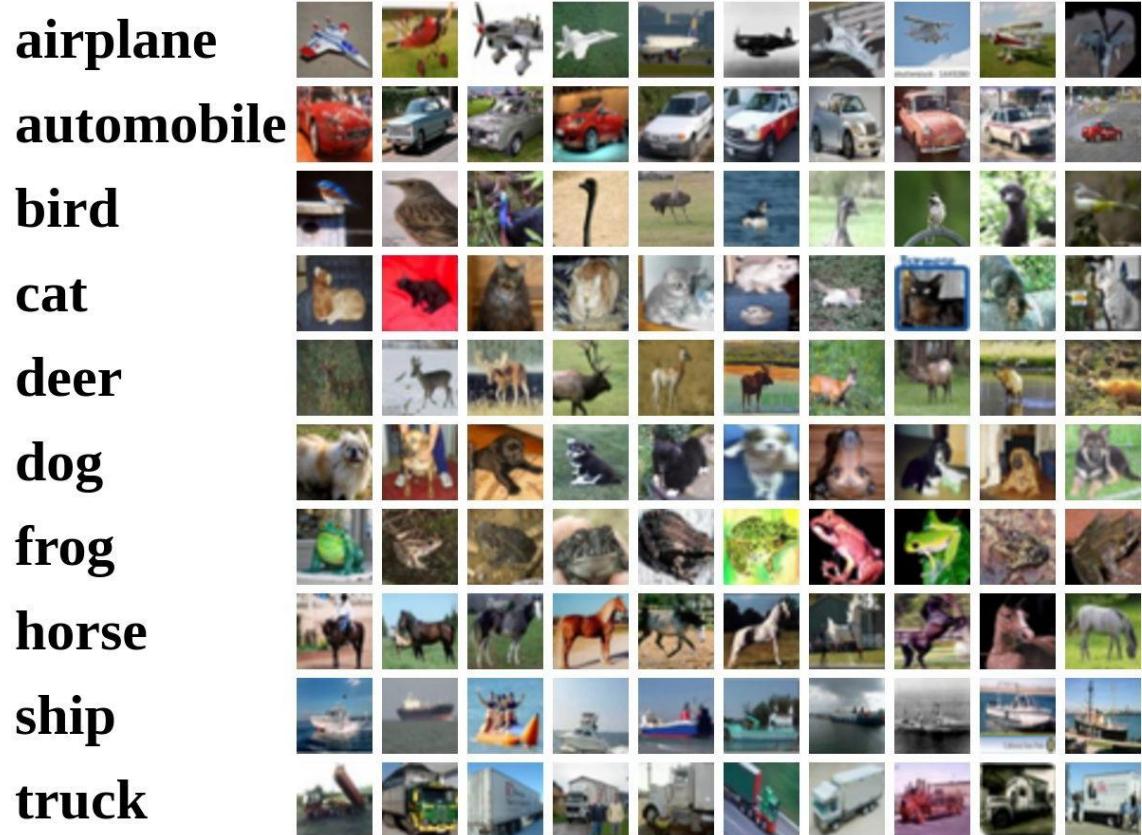
— Trong đó:

+ x : ảnh.

+ W : ma trận trọng số.

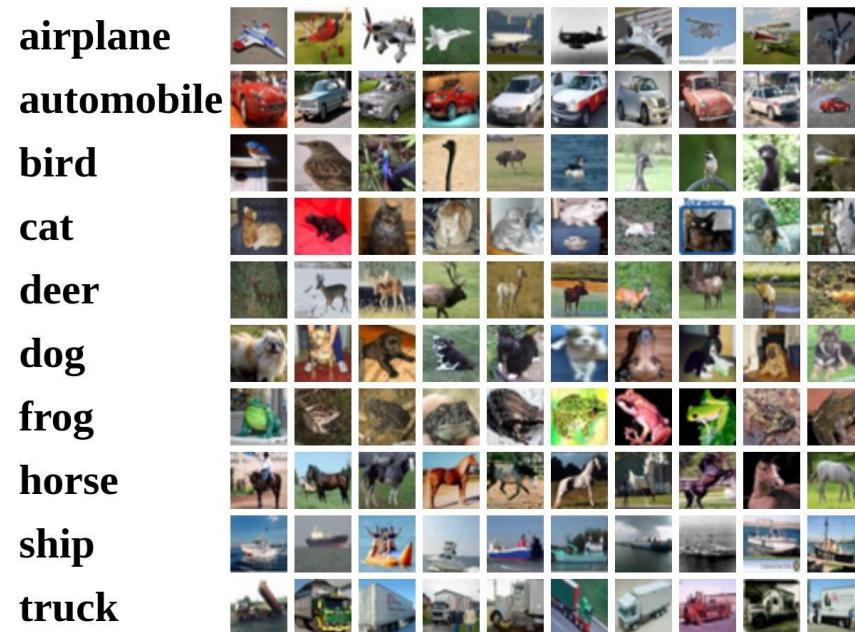
+ b : hệ số *bias*.

+ W, b : model, classifier.



Parametric Approach: Linear Classifier

— Ma trận trọng số W :



$$\begin{matrix} w_{0,0} & w_{0,1} & \dots & w_{0,3070} & w_{0,3071} \\ w_{1,0} & w_{1,1} & \dots & w_{1,3070} & w_{1,3071} \\ \dots & \dots & \dots & \dots & \dots \\ w_{8,0} & w_{8,1} & \dots & w_{8,3070} & w_{8,3071} \\ w_{9,0} & w_{9,1} & \dots & w_{9,3070} & w_{9,3071} \end{matrix}$$

Parametric Approach: Linear Classifier

- Mô hình Linear Classifier:

$$f(x, W) = W \cdot x + b = score$$

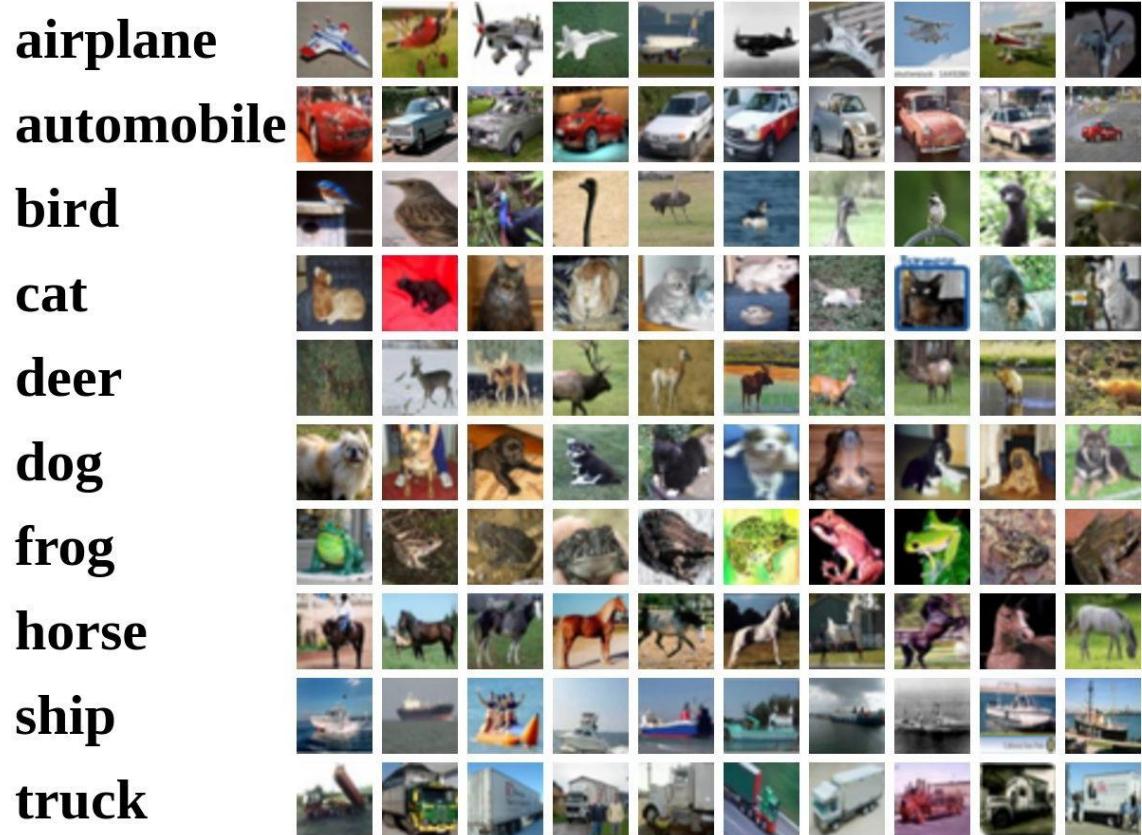
- Trong đó:

- + x : ảnh.

- + W : ma trận trọng số.

- + b : hệ số *bias*.

- + W, b : model, classifier.

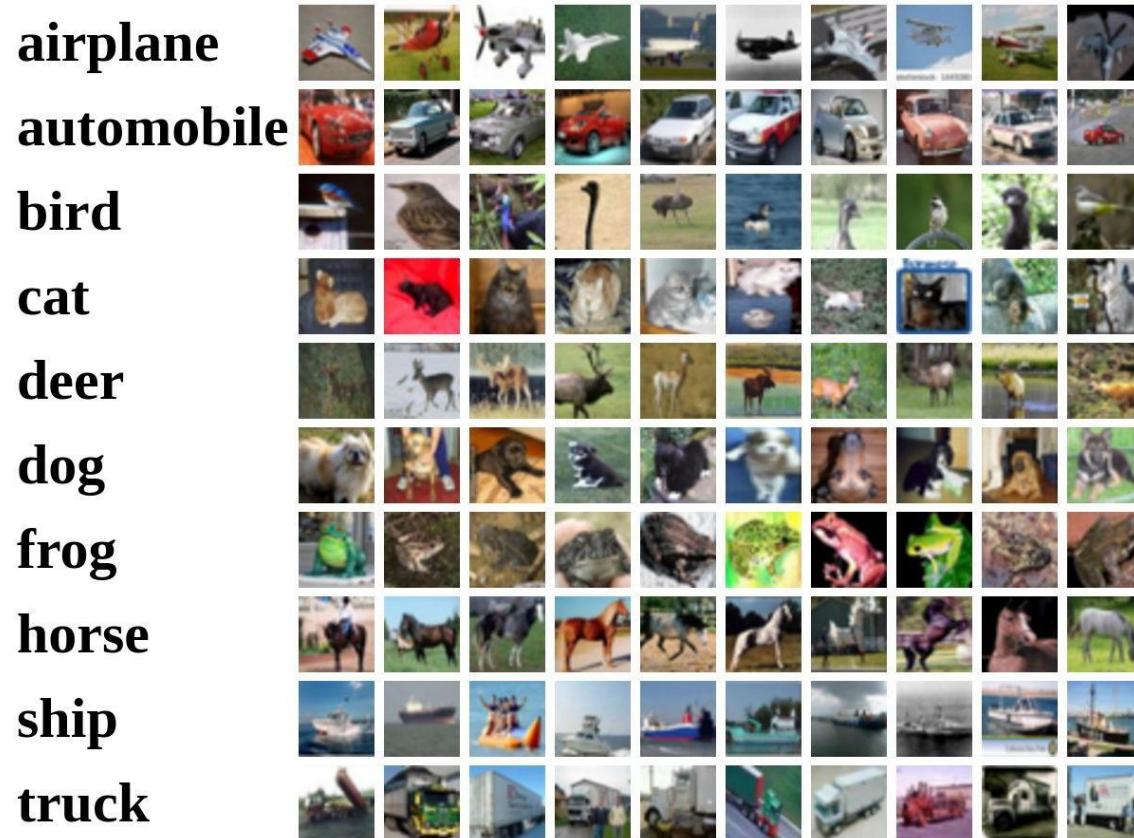


Parametric Approach: Linear Classifier

$$\begin{pmatrix}
w_{0,0} & w_{0,1} & \dots & w_{0,3070} & w_{0,3071} \\
w_{1,0} & w_{1,1} & \dots & w_{1,3070} & w_{1,3071} \\
\dots & \dots & \dots & \dots & \dots \\
w_{8,0} & w_{8,1} & \dots & w_{8,3070} & w_{8,3071} \\
w_{9,0} & w_{9,1} & \dots & w_{9,3070} & w_{9,3071}
\end{pmatrix}
\begin{pmatrix}
x_0 \\
x_1 \\
\dots \\
x_i \\
\dots \\
x_{3069} \\
x_{3070} \\
x_{3071}
\end{pmatrix}
+
\begin{pmatrix}
b_0 \\
b_1 \\
\dots \\
b_8 \\
b_9
\end{pmatrix}
=
\begin{pmatrix}
s_0 \\
s_1 \\
\dots \\
s_8 \\
s_9
\end{pmatrix}$$

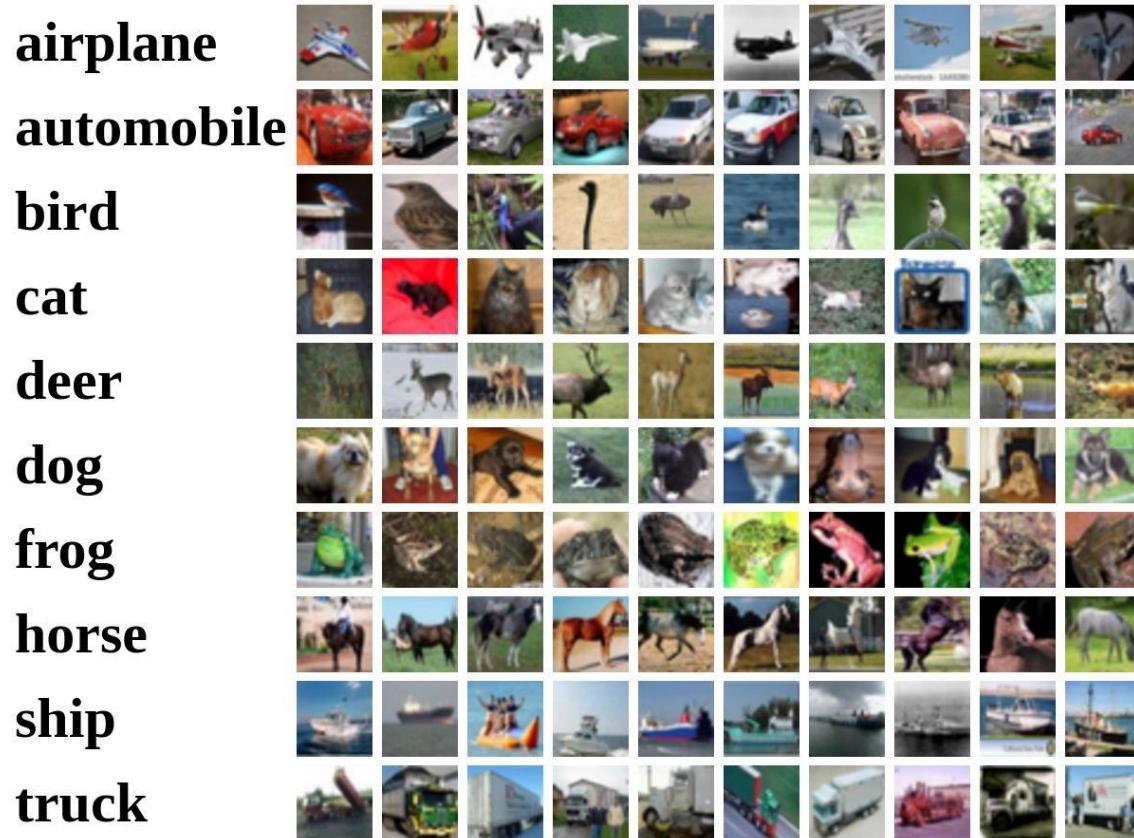
10×1 10×3072 10×1
 $f(\mathbf{x}, \mathbf{W})$ = $\mathbf{W} \otimes$ + \mathbf{b}
3072 × 1

Parametric Approach: Linear Classifier



- Mô hình Linear Classifier:
- $$f(x, W) = Wx + b$$
- What is this thing doing?

Parametric Approach: Linear Classifier



- Mô hình Linear Classifier:

$$f(x, W) = Wx + b$$

- Example trained weights of a linear classifier trained on CIFAR-10:

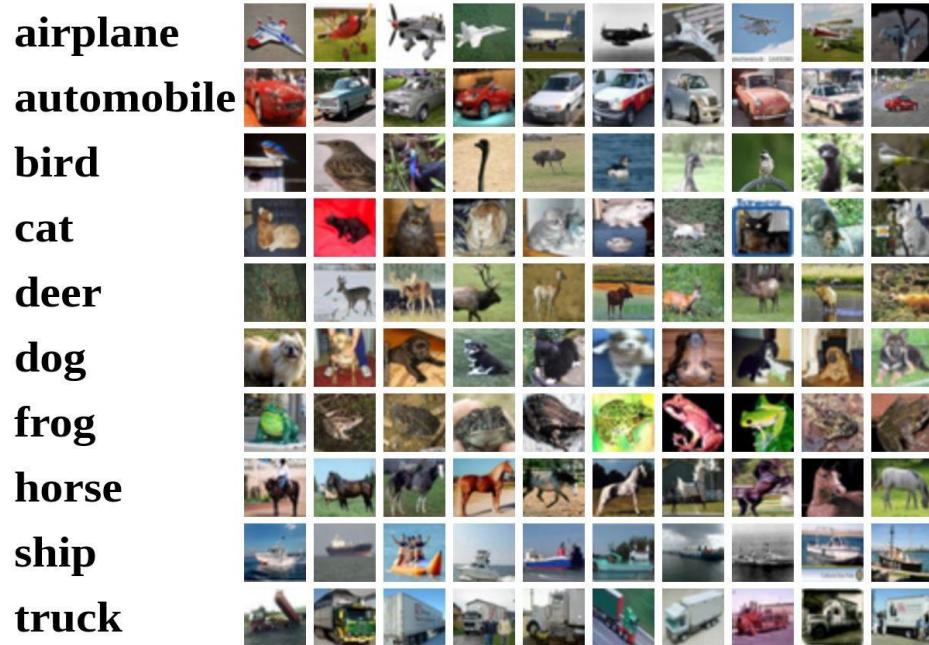
- + 50,000 training images each image is $32 \times 32 \times 3$.

- + 10,000 testing images each image is $32 \times 32 \times 3$.

Parametric Approach: Linear Classifier

$$\begin{pmatrix}
w_{0,0} & w_{0,1} & \dots & w_{0,3070} & w_{0,3071} \\
w_{1,0} & w_{1,1} & \dots & w_{1,3070} & w_{1,3071} \\
\dots & \dots & \dots & \dots & \dots \\
w_{8,0} & w_{8,1} & \dots & w_{8,3070} & w_{8,3071} \\
w_{9,0} & w_{9,1} & \dots & w_{9,3070} & w_{9,3071}
\end{pmatrix}
\begin{pmatrix}
x_0^{(i)} \\
x_1^{(i)} \\
\dots \\
x_i^{(i)} \\
\dots \\
x_{3069}^{(i)} \\
x_{3070}^{(i)} \\
x_{3071}^{(i)}
\end{pmatrix}
+
\begin{pmatrix}
b_0 \\
b_1 \\
\dots \\
b_8 \\
b_9
\end{pmatrix}
=
\begin{pmatrix}
s_0^{(i)} \\
s_1^{(i)} \\
\dots \\
s_8^{(i)} \\
s_9^{(i)}
\end{pmatrix}$$

Parametric Approach: Linear Classifier



– Mô hình Linear Classifier:

$$f(x, W) = Wx + b$$

– Example trained weights of a linear classifier trained on CIFAR-10:

Parametric Approach: Linear Classifier

$$\begin{aligned}
 & \left(\begin{array}{cccc} w_{0,0} & w_{0,1} & \dots & w_{0,3070} & w_{0,3071} \\ w_{1,0} & w_{1,1} & \dots & w_{1,3070} & w_{1,3071} \\ \dots & \dots & \dots & \dots & \dots \\ w_{8,0} & w_{8,1} & \dots & w_{8,3070} & w_{8,3071} \\ w_{9,0} & w_{9,1} & \dots & w_{9,3070} & w_{9,3071} \end{array} \right) \\
 & \quad \left(\begin{array}{c} 10 \times 1 \\ 10 \times 3072 \\ \boxed{f(\mathbf{x}, \mathbf{W})} \end{array} \right) = \left(\begin{array}{c} \mathbf{W} \otimes \\ \mathbf{x} \end{array} \right) + \left(\begin{array}{c} \mathbf{b} \\ 3072 \times 1 \end{array} \right) \\
 & \quad \left(\begin{array}{c} x_0^{(i)} \\ x_1^{(i)} \\ \dots \\ x_i^{(i)} \\ \dots \\ x_{3069}^{(i)} \\ x_{3070}^{(i)} \\ x_{3071}^{(i)} \end{array} \right) + \left(\begin{array}{c} b_0 \\ b_1 \\ \dots \\ b_8 \\ b_9 \end{array} \right) = \left(\begin{array}{c} s_0^{(i)} \\ s_1^{(i)} \\ \dots \\ s_8^{(i)} \\ s_9^{(i)} \end{array} \right)
 \end{aligned}$$

Parametric Approach: Linear Classifier

$$\begin{pmatrix}
 w_{0,0} & w_{0,1} & \dots & w_{0,3070} & w_{0,3071} \\
 w_{1,0} & w_{1,1} & \dots & w_{1,3070} & w_{1,3071} \\
 \dots & \dots & \dots & \dots & \dots \\
 w_{8,0} & w_{8,1} & \dots & w_{8,3070} & w_{8,3071} \\
 w_{9,0} & w_{9,1} & \dots & w_{9,3070} & w_{9,3071}
 \end{pmatrix}
 \begin{pmatrix}
 x_0^{(i)} \\
 x_1^{(i)} \\
 \dots \\
 x_i^{(i)} \\
 \dots \\
 x_{3069}^{(i)} \\
 x_{3070}^{(i)} \\
 x_{3071}^{(i)}
 \end{pmatrix}
 + \begin{pmatrix}
 b_0 \\
 b_1 \\
 \dots \\
 b_8 \\
 b_9
 \end{pmatrix} = \begin{pmatrix}
 s_0^{(i)} \\
 s_1^{(i)} \\
 \dots \\
 s_8^{(i)} \\
 s_9^{(i)}
 \end{pmatrix}$$

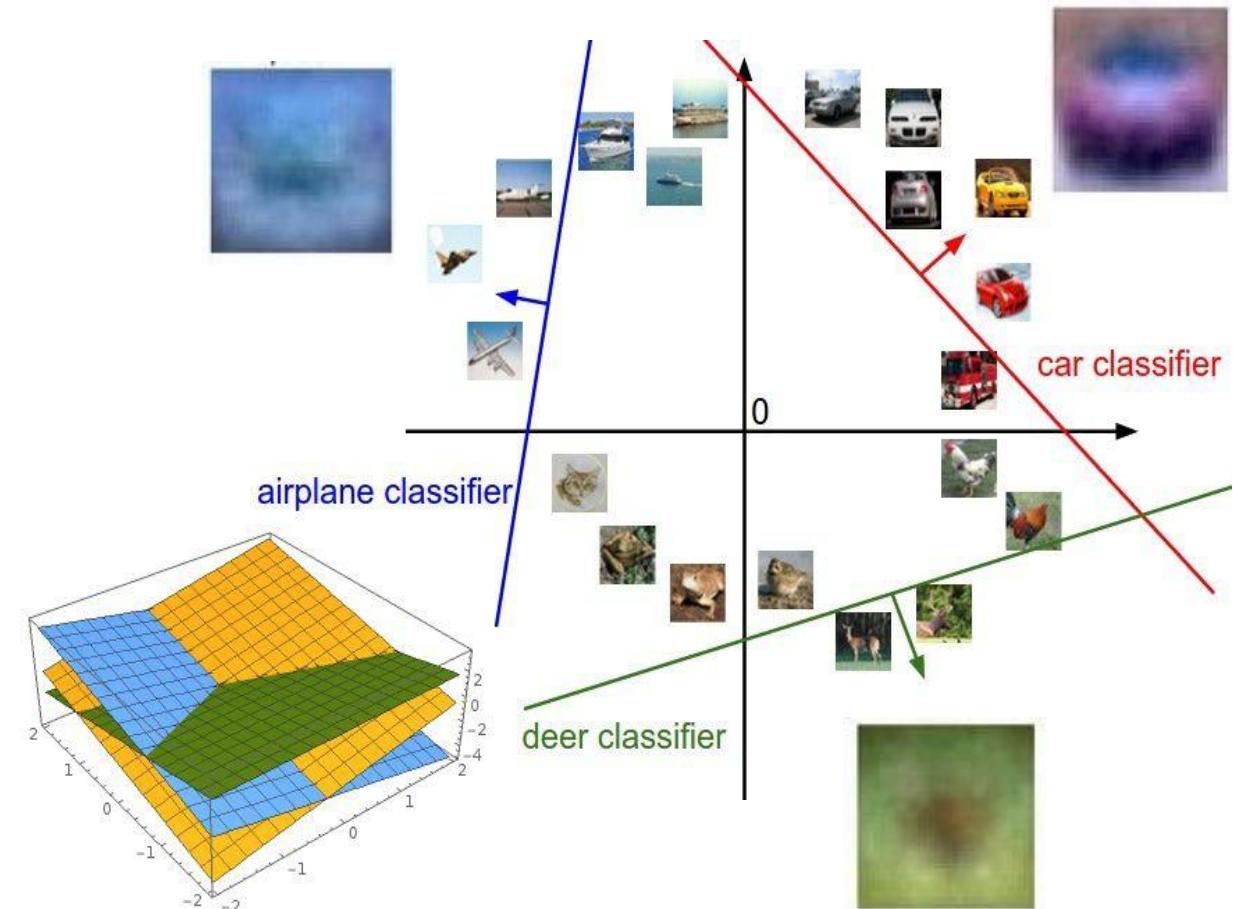
Parametric Approach: Linear Classifier

$$\begin{pmatrix}
 w_{0,0} & w_{0,1} & \dots & w_{0,3070} & w_{0,3071} & b_0 \\
 w_{1,0} & w_{1,1} & \dots & w_{1,3070} & w_{1,3071} & b_1 \\
 \dots & \dots & \dots & \dots & \dots & \dots \\
 w_{8,0} & w_{8,1} & \dots & w_{8,3070} & w_{8,3071} & b_8 \\
 w_{9,0} & w_{9,1} & \dots & w_{9,3070} & w_{9,3071} & b_9
 \end{pmatrix}
 \begin{pmatrix}
 x_0^{(i)} \\ x_1^{(i)} \\ \dots \\ x_i^{(i)} \\ \dots \\ x_{3069}^{(i)} \\ x_{3070}^{(i)} \\ x_{3071}^{(i)} \\ 1
 \end{pmatrix}
 = \begin{pmatrix}
 s_0^{(i)} \\ s_1^{(i)} \\ \dots \\ s_8^{(i)} \\ s_9^{(i)}
 \end{pmatrix}$$

Parametric Approach: Linear Classifier

$$\begin{array}{c}
 \begin{matrix} 10 \times 1 & 10 \times 3073 \\ f(\mathbf{x}, \mathbf{W}) & = \mathbf{W} \otimes \end{matrix} \\
 \begin{matrix} 3073 \times 1 \\ \left(\begin{matrix} w_{0,0} & w_{0,1} & \dots & w_{0,3070} & w_{0,3071} & b_0 \\ w_{1,0} & w_{1,1} & \dots & w_{1,3070} & w_{1,3071} & b_1 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ w_{8,0} & w_{8,1} & \dots & w_{8,3070} & w_{8,3071} & b_8 \\ w_{9,0} & w_{9,1} & \dots & w_{9,3070} & w_{9,3071} & b_9 \end{matrix} \right) \end{matrix} \\
 \begin{matrix} x_0^{(i)} \\ x_1^{(i)} \\ \dots \\ x_i^{(i)} \\ \dots \\ x_{3069}^{(i)} \\ x_{3070}^{(i)} \\ x_{3071}^{(i)} \\ 1 \end{matrix} = \begin{pmatrix} s_0^{(i)} \\ s_1^{(i)} \\ \dots \\ s_8^{(i)} \\ s_9^{(i)} \end{pmatrix}
 \end{array}$$

Parametric Approach: Linear Classifier

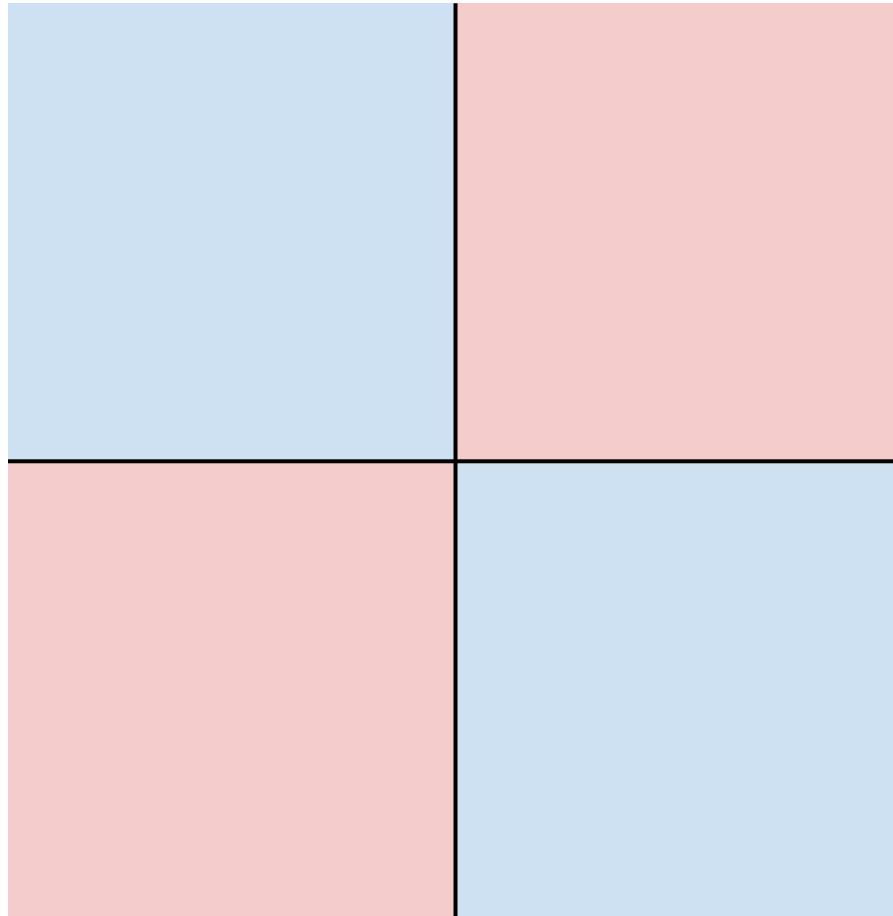


$$f(x, W) = Wx + b$$



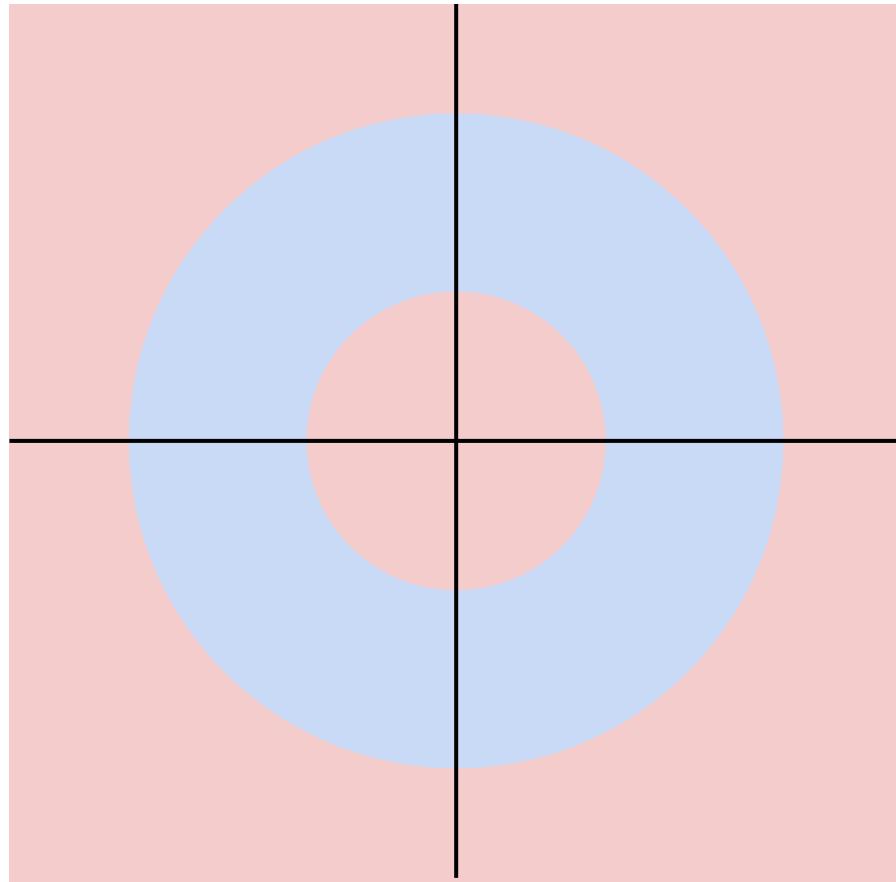
– Array of $32 \times 32 \times 3$ numbers
(3,072 numbers total)

Hard cases for a linear classifier



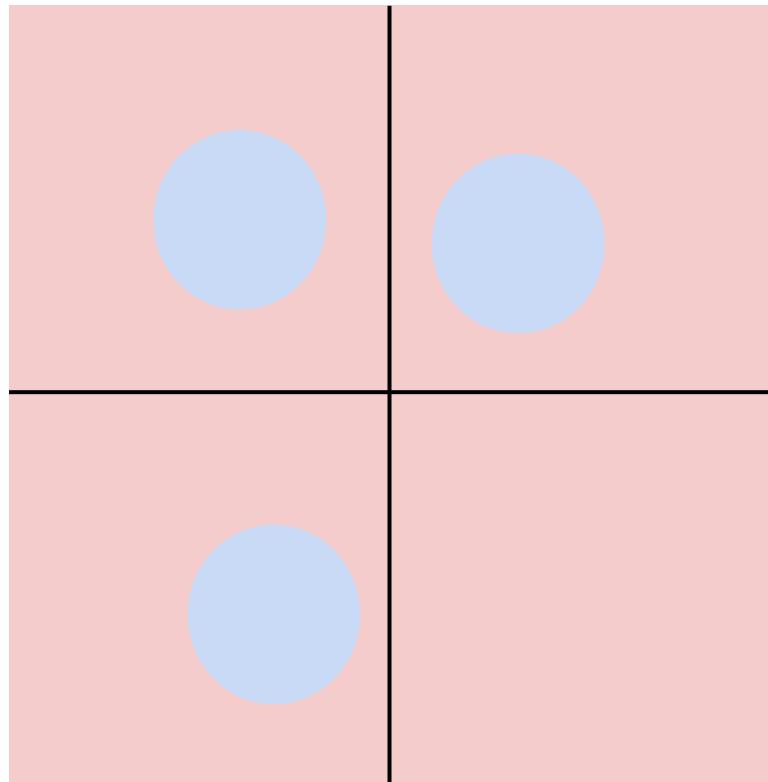
- Class 1:
 - + number of pixels > 0 odd
- Class 2:
 - + number of pixels > 0 even

Hard cases for a linear classifier



- Class 1:
 - + $1 \leq \text{L2 norm} \leq 2$
- Class 2:
 - + Everything else

Hard cases for a linear classifier



- Class 1:
 - + Three modes
- Class 2:
 - + Everything else

W is good or bad?



| | Cat | Automobile | Frog |
|------------|-------|------------|-------|
| Airplane | -3.45 | -0.51 | 3.42 |
| Automobile | -8.87 | 6.04 | 4.64 |
| Bird | 0.09 | 5.31 | 2.65 |
| Cat | 2.90 | -4.22 | 5.10 |
| Deer | 4.48 | -4.19 | 2.64 |
| Dog | 8.02 | 3.58 | 5.55 |
| Frog | 3.78 | 4.49 | -4.34 |
| Horse | 1.06 | -4.37 | -1.50 |
| Ship | -0.36 | -2.09 | -4.79 |
| Truck | -0.72 | -2.93 | 6.14 |

- Example class scores for 3 images for some W:

- How can we tell whether this W is good or bad?

W is good or bad?



| | Cat | Automobile | Frog |
|------------|-------|------------|-------|
| Airplane | -3.45 | -0.51 | 3.42 |
| Automobile | -8.87 | 6.04 | 4.64 |
| Bird | 0.09 | 5.31 | 2.65 |
| Cat | 2.90 | -4.22 | 5.10 |
| Deer | 4.48 | -4.19 | 2.64 |
| Dog | 8.02 | 3.58 | 5.55 |
| Frog | 3.78 | 4.49 | -4.34 |
| Horse | 1.06 | -4.37 | -1.50 |
| Ship | -0.36 | -2.09 | -4.79 |
| Truck | -0.72 | -2.93 | 6.14 |

- Example class scores for 3 images for some W:

- How can we tell whether this W is good or bad?

W is good or bad?



| | Cat | Automobile | Frog |
|------------|-------|------------|-------|
| Airplane | -3.45 | -0.51 | 3.42 |
| Automobile | -8.87 | 6.04 | 4.64 |
| Bird | 0.09 | 5.31 | 2.65 |
| Cat | 2.90 | -4.22 | 5.10 |
| Deer | 4.48 | -4.19 | 2.64 |
| Dog | 8.02 | 3.58 | 5.55 |
| Frog | 3.78 | 4.49 | -4.34 |
| Horse | 1.06 | -4.37 | -1.50 |
| Ship | -0.36 | -2.09 | -4.79 |
| Truck | -0.72 | -2.93 | 6.14 |

- Example class scores for 3 images for some W:

- How can we tell whether this W is good or bad?

W is good or bad?



| | Cat | Automobile | Frog |
|------------|-------|------------|-------|
| Airplane | -3.45 | -0.51 | 3.42 |
| Automobile | -8.87 | 6.04 | 4.64 |
| Bird | 0.09 | 5.31 | 2.65 |
| Cat | 2.90 | -4.22 | 5.10 |
| Deer | 4.48 | -4.19 | 2.64 |
| Dog | 8.02 | 3.58 | 5.55 |
| Frog | 3.78 | 4.49 | -4.34 |
| Horse | 1.06 | -4.37 | -1.50 |
| Ship | -0.36 | -2.09 | -4.79 |
| Truck | -0.72 | -2.93 | 6.14 |

- Example class scores for 3 images for some W:

- How can we tell whether this W is good or bad?

W is good or bad?



| | Cat | Automobile | Frog |
|------------|-------|------------|-------|
| Airplane | -3.45 | -0.51 | 3.42 |
| Automobile | -8.87 | 6.04 | 4.64 |
| Bird | 0.09 | 5.31 | 2.65 |
| Cat | 2.90 | -4.22 | 5.10 |
| Deer | 4.48 | -4.19 | 2.64 |
| Dog | 8.02 | 3.58 | 5.55 |
| Frog | 3.78 | 4.49 | -4.34 |
| Horse | 1.06 | -4.37 | -1.50 |
| Ship | -0.36 | -2.09 | -4.79 |
| Truck | -0.72 | -2.93 | 6.14 |

- Example class scores for 3 images for some W:

- How can we tell whether this W is good or bad?

W is good or bad?



| | Cat | Automobile | Frog |
|------------|-------|------------|-------|
| Airplane | -3.45 | -0.51 | 3.42 |
| Automobile | -8.87 | 6.04 | 4.64 |
| Bird | 0.09 | 5.31 | 2.65 |
| Cat | 2.90 | -4.22 | 5.10 |
| Deer | 4.48 | -4.19 | 2.64 |
| Dog | 8.02 | 3.58 | 5.55 |
| Frog | 3.78 | 4.49 | -4.34 |
| Horse | 1.06 | -4.37 | -1.50 |
| Ship | -0.36 | -2.09 | -4.79 |
| Truck | -0.72 | -2.93 | 6.14 |

- Example class scores for 3 images for some W:

- How can we tell whether this W is good or bad?

W is good or bad?



| | Cat | Automobile | Frog |
|------------|-------|------------|-------|
| Airplane | -3.45 | -0.51 | 3.42 |
| Automobile | -8.87 | 6.04 | 4.64 |
| Bird | 0.09 | 5.31 | 2.65 |
| Cat | 2.90 | -4.22 | 5.10 |
| Deer | 4.48 | -4.19 | 2.64 |
| Dog | 8.02 | 3.58 | 5.55 |
| Frog | 3.78 | 4.49 | -4.34 |
| Horse | 1.06 | -4.37 | -1.50 |
| Ship | -0.36 | -2.09 | -4.79 |
| Truck | -0.72 | -2.93 | 6.14 |

- Example class scores for 3 images for some W:

- How can we tell whether this W is good or bad?

W is good or bad?



| | Cat | Automobile | Frog |
|------------|-------|------------|-------|
| Airplane | -3.45 | -0.51 | 3.42 |
| Automobile | -8.87 | 6.04 | 4.64 |
| Bird | 0.09 | 5.31 | 2.65 |
| Cat | 2.90 | -4.22 | 5.10 |
| Deer | 4.48 | -4.19 | 2.64 |
| Dog | 8.02 | 3.58 | 5.55 |
| Frog | 3.78 | 4.49 | -4.34 |
| Horse | 1.06 | -4.37 | -1.50 |
| Ship | -0.36 | -2.09 | -4.79 |
| Truck | -0.72 | -2.93 | 6.14 |

- Example class scores for 3 images for some W:

- How can we tell whether this W is good or bad?

W is good or bad?



| | Cat | Automobile | Frog |
|------------|-------|------------|-------|
| Airplane | -3.45 | -0.51 | 3.42 |
| Automobile | -8.87 | 6.04 | 4.64 |
| Bird | 0.09 | 5.31 | 2.65 |
| Cat | 2.90 | -4.22 | 5.10 |
| Deer | 4.48 | -4.19 | 2.64 |
| Dog | 8.02 | 3.58 | 5.55 |
| Frog | 3.78 | 4.49 | -4.34 |
| Horse | 1.06 | -4.37 | -1.50 |
| Ship | -0.36 | -2.09 | -4.79 |
| Truck | -0.72 | -2.93 | 6.14 |

- Example class scores for 3 images for some W:

- How can we tell whether this W is good or bad?

—Bad

Coming up

$$f(x, W) = Wx + b$$

- Loss function (quantifying what it means to have a “good” W).
- Optimization (start with random W and find a W that minimizes the loss).
- ConvNets! (tweak the functional form of f).

BÀI TẬP LINEAR CLASSIFIER

Bài tập linear classifier

— Bài tập 21:

- + Load bộ dữ liệu uitTinyTrainCIFAR10 trong thư mục làm việc của Colab.
- + Phát sinh ngẫu nhiên ma trận W .
- + Tính điểm cho từng điểm dữ liệu.
- + Tính độ chính xác.

Chúc các bạn học tốt
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN TP.HCM

Nhóm UIT-Together
TS. Nguyễn Tân Trần Minh Khang

Generating Image Descriptions



- Two young girls are playing with lego toy.
- Hai cô gái trẻ đang chơi đồ chơi Lego.

Karpathy and Fei-Fei, Deep Visual-Semantic Alignments for Generating Image Descriptions, CVPR 2015

Generating Image Descriptions



- Boy is doing backflip on wakeboard.
- Boy is doing backflip on wakeboard" có nghĩa là "Cậu bé đang thực hiện động tác lộn ngược trên ván lướt sóng có dây kéo.

Karpathy and Fei-Fei, Deep Visual-Semantic Alignments for Generating Image Descriptions, CVPR 2015

Generating Image Descriptions



- Man in black shirt is playing guitar.
- Người đàn ông mặc áo đen đang chơi guitar.

Karpathy and Fei-Fei, Deep Visual-Semantic Alignments for Generating Image Descriptions, CVPR 2015

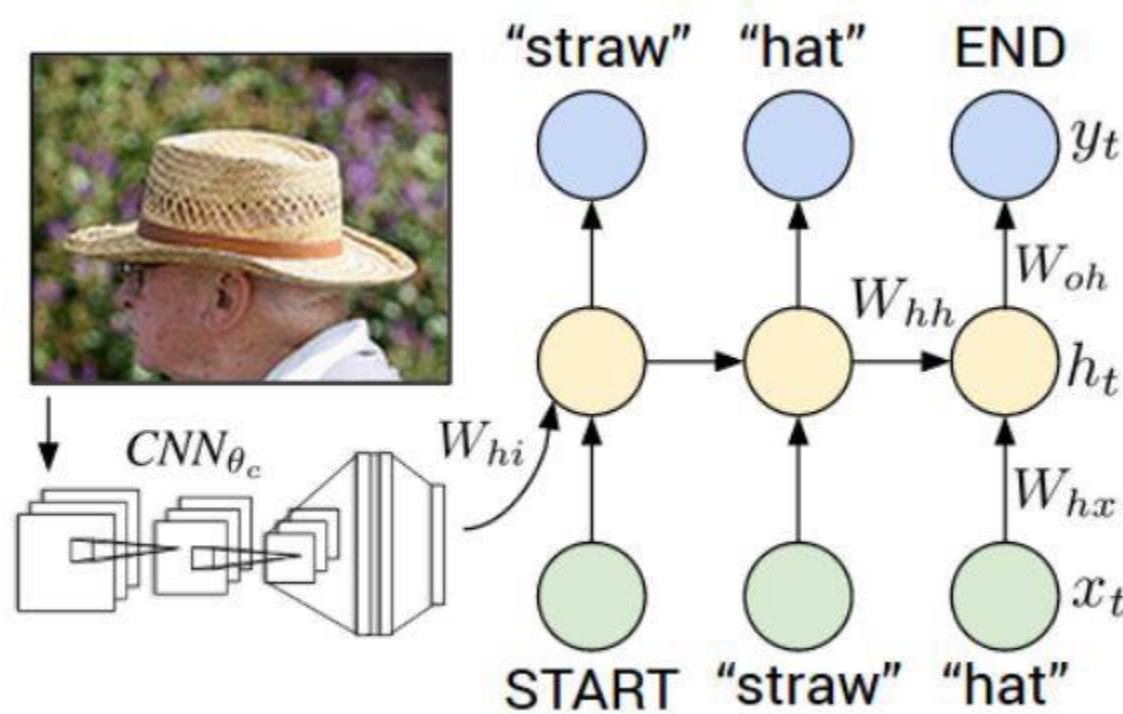
Generating Image Descriptions



- Construction worker in orange safety vest is working on road.
- Công nhân xây dựng mặc áo bảo hộ màu cam đang làm việc trên đường.

Karpathy and Fei-Fei, Deep Visual-Semantic Alignments for Generating Image Descriptions, CVPR 2015

Generating Image Descriptions



Karpathy and Fei-Fei, Deep Visual-Semantic Alignments for Generating Image Descriptions, CVPR 2015

BÀI TẬP CIFAR 10

Bài tập CIFAR 10

- Bài tập 01:
 - + Load bộ dữ liệu CIFAR10 theo nguyên bản của đại học Toronto về thư mục làm việc của Colab.

Bài tập 01

11.import os

12.import pickle

13.import numpy as np

Bài tập 01

```
14.def load_CIFAR_batch(filename):  
15.    with open(filename, 'rb') as f:  
16.        datadict = pickle.load(f, encoding='bytes')  
17.        X = datadict[b'data']  
18.        y = datadict[b'labels']  
19.        y = np.array(y)  
20.        return X, y
```

Bài tập 01

```
21. def load_CIFAR10(directory):  
22.     x_train = []  
23.     y_train = []  
24.     for i in range(1, 6):  
25.         filename      =          os.path.join(directory,  
'data_batch_%d' % (i,))  
26.         X, y = load_CIFAR_batch(filename)  
27.         X_train.append(X)  
28.         y_train.append(y)
```

Bài tập 01

```
29.     X_train = np.concatenate(X_train)
30.     y_train = np.concatenate(y_train)
31.     X_test,y_test=
32.         load_CIFAR_batch(os.path.join(directory,
33.             'test_batch'))
33.     return X_train, y_train, X_test, y_test
```

```
34.cifar_directory = './cifar-10-batches-py/'  
35.X_train,      y_train,      X_test,      y_test      =  
    load_CIFAR10(cifar_directory)  
36.print("Training data shape: ", X_train.shape)  
37.print("Training labels shape: ", y_train.shape)  
38.print("Test data shape: ", X_test.shape)  
39.print("Test labels shape: ", y_test.shape)
```

Bài tập CIFAR 10

- Bài tập 02:
 - + Load bộ dữ liệu CIFAR10 trong thư mục làm việc của Colab.
 - + Đếm số lượng điểm dữ liệu theo từng lớp nhãn trong bộ dữ liệu train nguyên bản.
 - + Đếm số lượng điểm dữ liệu theo từng lớp nhãn trong bộ dữ liệu test nguyên bản.
 - + Vẽ biểu đồ minh họa số lượng điểm dữ liệu theo từng lớp nhãn trong bộ dữ liệu (train và test).

Bài tập 02

11. import os

12. import pickle

13. import numpy as np

Bài tập 02

```
14.def load_CIFAR_batch(filename):  
15.    with open(filename, 'rb') as f:  
16.        datadict = pickle.load(f, encoding='bytes')  
17.        X = datadict[b'data']  
18.        y = datadict[b'labels']  
19.        y = np.array(y)  
20.        return X, y
```

Bài tập 02

```
21.def DemSoLuongAnh1Batch(data_batch_file):  
22.    # Nạp dữ liệu từ file data_batch  
23.    X, Y = load_CIFAR_batch(data_batch_file)  
24.    # Dictionary để đếm số lượng ảnh cho mỗi lớp  
25.    class_counts = {}
```

Bài tập 02

```
26.     for label in Y:  
27.         if label in class_counts:  
28.             class_counts[label] += 1  
29.         else:  
30.             class_counts[label] = 1  
31.     return class_counts
```

Bài tập 02

```
32. def DemSoLuongAnhTatCaBatch(directory):  
33.     total_class_counts = {}  
34.     # Đếm số lượng ảnh từ các batch huấn luyện  
35.     all_batches_counts = []  
36.     for i in range(1, 6):  
37.         filename=os.path.join(directory,  
'data_batch_%d' % (i,))  
38.         batch_counts = DemSoLuongAnh1Batch(filename)  
39.         all_batches_counts.append((i, batch_counts))
```

Bài tập 02

```
32.     for class_label, count in batch_counts.items():
33.         if class_label in total_class_counts:
34.             total_class_counts[class_label] += count
35.         else:
36.             total_class_counts[class_label] = count
37.     return total_class_counts, all_batches_counts
```

Bài tập 02

1. # Đường dẫn đến thư mục chứa dữ liệu CIFAR-10
2. cifar_directory = './cifar-10-batches-py/'
3. # Đếm số lượng ảnh từ tất cả các batch
4. total_class_counts, all_batches_counts= DemSoLuongAnhTatCaBatch(cifar_directory)

Bài tập 02

```
1. y = ['airplane', 'automobile', 'bird', 'cat',
'deer', 'dog', 'frog', 'horse', 'ship', 'truck']
2. print("all batch counts: ", all_batches_counts)
3. # In số lượng ảnh theo từng lớp trong từng batch
huấn luyện
4. for batch_num, batch_counts in all_batches_counts:
5.     print(f'data_batch_{batch_num}:')
6.     for class_label in sorted(batch_counts.keys()):
7.         print(f'{y[class_label]}:
{batch_counts[class_label]} images')
```

Bài tập 02

1. # Đếm số lượng ảnh từ tất cả các batch
2. total_class_counts, all_batches_counts =
DemSoLuongAnhTatCaBatch(cifar_directory)

3. # In tổng số lượng ảnh theo từng lớp trên toàn bộ bộ dữ liệu huấn luyện
4. print('Total counts across all training batches:')
5. for class_label in sorted(total_class_counts.keys()):
6. print(f'{y[class_label]}:
{total_class_counts[class_label]} images')

Bài tập 02

```
1. test_batch_file      =      os.path.join(cifar_directory,  
'test_batch')  
2. test_batch_counts          =  
DemSoLuongAnh1Batch(test_batch_file)  
3. # In số lượng ảnh theo từng lớp trong batch kiểm tra  
4. print('test_batch: ')  
5. for class_label in sorted(test_batch_counts.keys()):  
6.     print(f'{y[class_label]}:  
{test_batch_counts[class_label]} images')
```

Bài tập 02

```
1. colors = ['b', 'g', 'r', 'c', 'm', 'y']
2. # Vẽ biểu đồ số lượng ảnh theo từng lớp trong từng
batch huấn luyện
3. for batch_num, batch_counts in train_batches_counts:
4.     sorted_counts = [batch_counts.get(i, 0) for i in
range(len(y))]
5.     plt.figure(figsize=(10, 5))
```

Bài tập 02

```
1.     plt.bar(y, sorted_counts, color=colors[batch_num  
% len(colors)], alpha=0.7)  
2.     plt.xlabel('Class')  
3.     plt.ylabel('Number of samples')  
4.     plt.title(f'Number of samples per class in  
data_batch_{batch_num}')  
5.     plt.show()
```

Bài tập 02

```
1. # Vẽ biểu đồ tổng số lượng ảnh trên bộ training set
2. sorted_train_counts = [total_class_counts.get(i, 0)
   for i in range(len(y))]
3. plt.figure(figsize=(10, 5))
4. plt.bar(y,sorted_train_counts, color='g', alpha=0.7)
5. plt.xlabel('Class')
6. plt.ylabel('Number of samples')
7. plt.title('Number images per class in training set')
8. plt.show()
```

Bài tập 02

```
1. # Biểu đồ số lượng ảnh theo từng lớp của test_batch
2. sorted_test_counts = [test_batch_counts.get(i, 0)
   for i in range(len(y))]
3. plt.figure(figsize=(10, 5))
4. plt.bar(y, sorted_test_counts, color='r', alpha=0.7)
5. plt.xlabel('Class')
6. plt.ylabel('Number of samples')
7. plt.title('Number images per class in test_batch')
8. plt.show()
```

Bài tập CIFAR 10

— Bài tập 03:

- + Load bộ dữ liệu CIFAR10 trong thư mục làm việc của Colab.
- + Chia bộ dữ liệu train có kích thước 50000 điểm dữ liệu thành hai bộ dữ liệu con có tên.
 - Bộ dữ liệu uitTrainCIFAR10 có kích thước 40000 điểm dữ liệu (số lượng điểm dữ liệu theo từng lớp nhãn bằng nhau).
 - Bộ dữ liệu uitValidationCIFAR10 có kích thước 10000.
- + Bộ dữ liệu test uitTestCIFAR10 có kích thước 10000 điểm dữ liệu (giống theo nguyên bản).

Bài tập 03

11. import os

12. import pickle

13. import numpy as np

Bài tập 03

```
14.def load_CIFAR_batch(filename):  
15.    with open(filename, 'rb') as f:  
16.        datadict = pickle.load(f, encoding='bytes')  
17.        X = datadict[b'data']  
18.        y = datadict[b'labels']  
19.        y = np.array(y)  
20.    return X, y
```

Bài tập 03

```
21. def load_CIFAR10(directory):  
22.     x_train = []  
23.     y_train = []  
24.     for i in range(1, 6):  
25.         filename      =          os.path.join(directory,  
'data_batch_%d' % (i,))  
26.         X, y = load_CIFAR_batch(filename)  
27.         X_train.append(X)  
28.         y_train.append(y)
```

Bài tập 03

```
29.     X_train = np.concatenate(X_train)
30.     y_train = np.concatenate(y_train)
31.     X_test,y_test=
32.         load_CIFAR_batch(os.path.join(directory,
33.             'test_batch'))
33.     return X_train, y_train, X_test, y_test
```

Bài tập 03

```
34.cifar_directory = './cifar-10-batches-py/'  
35.X_train,      y_train,      X_test,      y_test      =  
    load_CIFAR10(cifar_directory)  
36.print("Training data shape: ", X_train.shape)  
37.print("Training labels shape: ", y_train.shape)  
38.print("Test data shape: ", X_test.shape)  
39.print("Test labels shape: ", y_test.shape)
```

Bài tập 03

```
40. def split_train_val(x_train, y_train,  
        train_size_per_class, val_size_per_class,  
        num_classes=10, random_state=42):  
41.     x_train_final = []  
42.     y_train_final = []  
43.     x_val_final = []  
44.     y_val_final = []  
45.
```

Bài tập 03

```
40.     for i in range(num_classes):  
41.         index = np.where(y_train == i)[0]  
42.         X_class = X_train[index]  
43.         y_class = y_train[index]  
44.         X_class_train, X_class_val, y_class_train,  
y_class_val = train_test_split(X_class, y_class,  
train_size=train_size_per_class,  
test_size=val_size_per_class,  
random_state=random_state)
```

Bài tập 03

```
40.      X_train_final.append(X_class_train)
41.      y_train_final.append(y_class_train)
42.      X_val_final.append(X_class_val)
43.      y_val_final.append(y_class_val)
44.      X_train_final = np.concatenate(X_train_final)
45.      y_train_final = np.concatenate(y_train_final)
46.      X_val_final = np.concatenate(X_val_final)
47.      y_val_final = np.concatenate(y_val_final)
```

Bài tập 03

```
40.      X_train_final,y_train_final=  
        shuffle(X_train_final,y_train_final,  
random_state=random_state)  
40.      X_val_final, y_val_final = shuffle(X_val_final,  
y_val_final, random_state=random_state)  
41.      return          X_train_final,           y_train_final,  
X_val_final, y_val_final
```

Bài tập 03

```
42.train_size_per_class = 4000
43.val_size_per_class = 1000
44.num_classes = 10
45.X_train, y_train, X_val, y_val = split_train_val(
    X_train,y_train,train_size_per_class,val_size_per_cl
    ass,num_classes=num_classes,random_state=42)
46.print("New training data shape: ", X_train.shape)
47.print("New training labels shape: ", y_train.shape)
48.print("Validation data shape: ", X_val.shape)
49.print("Validation labels shape: ", y_val.shape)
```

Bài tập 03

```
50.import pickle
51.def save_cifar10_format(filename, data, labels):
52.    with open(filename, 'wb') as f:
53.        pickle.dump({
54.            b'batch_label': b'batch 1 of 1',
55.            b'labels': labels.tolist(),
56.            b'data': data.reshape(data.shape[0], -1)
... .astype(np.uint8),
57.            b'filenames': [b''] * data.shape[0]
58.        }, f)
```

Bài tập 03

```
51.save_cifar10_format('./dataset/uitTrainCIFAR10',  
    X_train, y_train)  
52.save_cifar10_format('./dataset/uitValidationCIFAR10,  
    X_val, y_val)
```

Bài tập 03

```
53.import shutil
54.src_path = './cifar-10-batches-py/test_batch'
55.dst_dir = './dataset'
56.dst_path = os.path.join(dst_dir, 'uitTestCIFAR10')
57.os.makedirs(dst_dir, exist_ok=True)
58.shutil.copy(src_path, dst_path)
```

Bài tập CIFAR 10

— Bài tập 04:

- + Load bộ dữ liệu uitTrainCIFAR10 trong thư mục làm việc của Colab.
- + Hiển thị 10 ảnh đầu tiên theo từng lớp nhãn.

airplane



automobile



bird



cat



deer



dog



frog



horse



ship



truck



Bài tập 04

Bài tập CIFAR 10

— Bài tập 05A:

- + Load bộ dữ liệu uitTrainCIFAR10 trong thư mục làm việc của Colab.
- + Tạo bộ dữ liệu uitTinyTrainCIFAR10 với mỗi lớp nhãn có 10 ảnh được lấy từ bộ dữ liệu uitTrainCIFAR10.

airplane



automobile



bird



cat



deer



dog



frog



horse



ship



truck



Bài tập 05A

1. import numpy as np
2. import os
3. import matplotlib.pyplot as plt
4. import cv2
5. import pickle

Bài tập 05A

```
14.def load_CIFAR_batch(filename):  
15.    with open(filename, 'rb') as f:  
16.        datadict = pickle.load(f, encoding='bytes')  
17.        X = datadict[b'data']  
18.        y = datadict[b'labels']  
19.        y = np.array(y)  
20.    return X, y
```

Bài tập 05A

1. # Định nghĩa hàm lưu batch theo số lượng ảnh theo từng class của bộ train
2. `def save_class_images_to_batch_train(x, y, classes, output_dir, num_images_per_class=10):`
3. `os.makedirs(output_dir, exist_ok=True)`
- 4.

Bài tập 05A

```
1.     if num_images_per_class == 10:  
2.         batch_name = "uitTinyTrainCIFAR10"  
3.     elif num_images_per_class == 100:  
4.         batch_name = "uitSmallTrainCIFAR10"  
5.     elif num_images_per_class == 1000:  
6.         batch_name = "uitMediumTrainCIFAR10"  
7.     elif num_images_per_class == 2000:  
8.         batch_name = "uitHalfTrainCIFAR10"
```

Bài tập 05A

```
1.     else:  
2.         batch_name          =  
3.         f"uit_{num_images_per_class}_TrainCIFAR10"  
4.         output_path        =      os.path.join(output_dir,  
5.             f"{batch_name}")  
6.         # Kiểm tra nếu batch đã tồn tại  
7.         if os.path.exists(output_path):  
8.             print(f"Batch {output_path} đã tồn tại.  
9.             Không ghi đè dữ liệu.")  
10.        return
```

Bài tập 05A

```
1.     X_new = []
2.     Y_new = []
3.     for idx, label in enumerate(classes):
4.         class_indices = np.where(y == idx)[0]
5.         # Lấy num_images_per_class ảnh ngẫu nhiên từ
       lớp này
6.         random_indices=
           np.random.choice(class_indices,
           num_images_per_class, replace=False)
```

Bài tập 05A

```
1.     class_images = X[random_indices]
2.         for img in class_images:
3.             X_new.append(img)
4.             Y_new.append(idx)
5. # Chuyển đổi thành numpy array
6. X_new = np.array(X_new)
7. Y_new = np.array(Y_new)
```

Bài tập 05A

```
1. # Tạo batch
2. batch = {
3.     b'data': X_new,
4.     b'labels': Y_new
5. }
6. # Lưu batch sử dụng pickle
7. with open(output_path, 'wb') as f:
8.     pickle.dump(batch, f)
9. print(f"Saved batch to {output_path}")
```

Bài tập 05A

1. # Đường dẫn dữ liệu và batch
2. data_train_path = '/content/drive/MyDrive/Bài tập CIFAR 10/dataset/uitTrainCIFAR10'
3. output_dir = '/content/drive/MyDrive/Bài tập CIFAR 10/dataset_split/tiny'
4. # Tạo tên các lớp nhãn
5. classes = ['airplane', 'automobile', 'bird', 'cat', 'deer', 'dog', 'frog', 'horse', 'ship', 'truck']

Bài tập 05A

1. # Load dữ liệu uitTrainCIFAR10
2. X_train, y_train = load_CIFAR_batch(data_train_path)
3. y_train = np.array(y_train)

4. # Lưu các ảnh ngẫu nhiên của mỗi lớp thành batch
5. save_class_images_to_batch_train(X_train, y_train, classes, output_dir, num_images_per_class=10)

Bài tập CIFAR 10

— Bài tập 05B:

+ Load bộ dữ liệu uitValidationCIFAR10 trong thư mục làm việc của Colab.

+ Tạo bộ dữ liệu uitTinyValidationCIFAR10 với mỗi lớp nhãn có 3 ảnh được lấy từ bộ dữ liệu uitValidationCIFAR10.

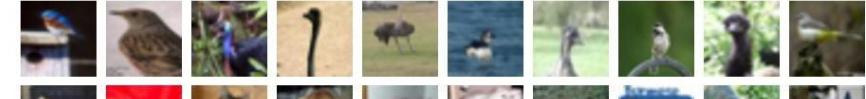
airplane



automobile



bird



cat



deer



dog



frog



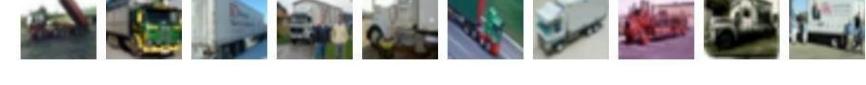
horse



ship



truck



Bài tập 05B

1. import numpy as np
2. import os
3. import matplotlib.pyplot as plt
4. import cv2
5. import pickle

Bài tập 05B

```
14. def load_CIFAR_batch(filename):  
15.     with open(filename, 'rb') as f:  
16.         datadict = pickle.load(f, encoding='bytes')  
17.         X = datadict[b'data']  
18.         y = datadict[b'labels']  
19.         y = np.array(y)  
20.     return X, y
```

Bài tập 05B

1. # Định nghĩa hàm lưu batch theo số lượng ảnh theo từng class của bộ val
2. def save_class_images_to_batch_validation(X, y, classes, output_dir, num_images_per_class=10):
3. os.makedirs(output_dir, exist_ok=True)
4.

Bài tập 05B

```
1.     if num_images_per_class == 3:  
2.         batch_name = "uitTinyValidationCIFAR10"  
3.     elif num_images_per_class == 25:  
4.         batch_name = "uitSmallValidationCIFAR10"  
5.     elif num_images_per_class == 250:  
6.         batch_name = "uitMediumValidationCIFAR10"  
7.     elif num_images_per_class == 500:  
8.         batch_name = "uitHalfValidationCIFAR10"
```

Bài tập 05B

```
1.     else:  
2.         batch_name =  
3.         output_path = os.path.join(output_dir,  
4.             f"{batch_name}")  
5.         # Kiểm tra nếu batch đã tồn tại  
6.         if os.path.exists(output_path):  
7.             print(f"Batch {output_path} đã tồn tại.  
        Không ghi đè dữ liệu.")  
8.         return
```

Bài tập 05B

```
1.     X_new = []
2.     Y_new = []
3.     for idx, label in enumerate(classes):
4.         class_indices = np.where(y == idx)[0]
5.         # Lấy num_images_per_class ảnh ngẫu nhiên từ
       lớp này
6.         random_indices=
           np.random.choice(class_indices,
           num_images_per_class, replace=False)
```

Bài tập 05B

```
1.     class_images = X[random_indices]
2.         for img in class_images:
3.             X_new.append(img)
4.             Y_new.append(idx)
5. # Chuyển đổi thành numpy array
6. X_new = np.array(X_new)
7. Y_new = np.array(Y_new)
```

Bài tập 05B

```
1. # Tạo batch
2. batch = {
3.     b'data': X_new,
4.     b'labels': Y_new
5. }
6. # Lưu batch sử dụng pickle
7. with open(output_path, 'wb') as f:
8.     pickle.dump(batch, f)
9. print(f"Saved batch to {output_path}")
```

Bài tập 05B

1. # Đường dẫn dữ liệu và batch
2. data_val_path = '/content/drive/MyDrive/Bài tập CIFAR 10/dataset/uitValidationCIFAR10'
3. output_dir = '/content/drive/MyDrive/Bài tập CIFAR 10/dataset_split/tiny'
4. # Tạo tên các lớp nhãn
5. classes = ['airplane', 'automobile', 'bird', 'cat', 'deer', 'dog', 'frog', 'horse', 'ship', 'truck']

Bài tập 05B

```
1. # Load dữ liệu uitValidationCIFAR10
2. X_val, y_val = load_CIFAR_batch(data_val_path)
3. y_val = np.array(y_val)
4. # Lưu các ảnh ngẫu nhiên của mỗi lớp thành batch
5. save_class_images_to_batch_validation(X_val, y_val,
   classes, output_dir, num_images_per_class=3)
```

Bài tập CIFAR 10

— Bài tập 05C:

- + Load bộ dữ liệu uitTestCIFAR10 trong thư mục làm việc của Colab.
- + Tạo bộ dữ liệu uitTinyTestCIFAR10 với mỗi lớp nhãn có 3 ảnh được lấy từ bộ dữ liệu uitTestCIFAR10.

airplane



automobile



bird



cat



deer



dog



frog



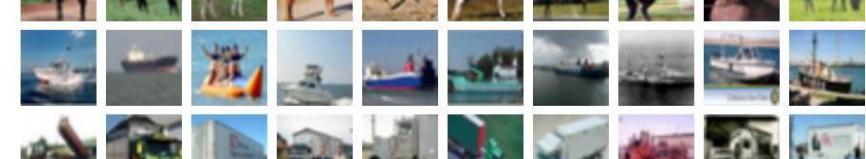
horse



ship



truck



Bài tập 05C

1. import numpy as np
2. import os
3. import matplotlib.pyplot as plt
4. import cv2
5. import pickle

Bài tập 05C

```
14.def load_CIFAR_batch(filename):  
15.    with open(filename, 'rb') as f:  
16.        datadict = pickle.load(f, encoding='bytes')  
17.        X = datadict[b'data']  
18.        y = datadict[b'labels']  
19.        y = np.array(y)  
20.    return X, y
```

Bài tập 05C

1. # Định nghĩa hàm lưu batch theo số lượng ảnh theo từng class của bộ test
2. `def save_class_images_to_batch_test(X, y, classes,
output_dir, num_images_per_class=10):`
3. `os.makedirs(output_dir, exist_ok=True)`
- 4.

Bài tập 05C

```
1.     if num_images_per_class == 3:  
2.         batch_name = "uitTinyTestCIFAR10"  
3.     elif num_images_per_class == 25:  
4.         batch_name = "uitSmallTestCIFAR10"  
5.     elif num_images_per_class == 250:  
6.         batch_name = "uitMediumTestCIFAR10"  
7.     elif num_images_per_class == 500:  
8.         batch_name = "uitHalfTestCIFAR10"
```

Bài tập 05C

```
1.     else:  
2.         batch_name           =  
3.         output_path          =      os.path.join(output_dir,  
4.             f"{batch_name}")  
5.         # Kiểm tra nếu batch đã tồn tại  
6.         if os.path.exists(output_path):  
7.             print(f"Batch {output_path} đã tồn tại.  
8.             Không ghi đè dữ liệu.")  
9.         return
```

Bài tập 05C

```
1.     X_new = []
2.     Y_new = []
3.     for idx, label in enumerate(classes):
4.         class_indices = np.where(y == idx)[0]
5.         # Lấy num_images_per_class ảnh ngẫu nhiên từ
       lớp này
6.         random_indices=
           np.random.choice(class_indices,
           num_images_per_class, replace=False)
```

Bài tập 05C

```
1.     class_images = X[random_indices]
2.         for img in class_images:
3.             X_new.append(img)
4.             Y_new.append(idx)
5. # Chuyển đổi thành numpy array
6. X_new = np.array(X_new)
7. Y_new = np.array(Y_new)
```

Bài tập 05C

```
1. # Tạo batch
2. batch = {
3.     b'data': X_new,
4.     b'labels': Y_new
5. }
6. # Lưu batch sử dụng pickle
7. with open(output_path, 'wb') as f:
8.     pickle.dump(batch, f)
9. print(f"Saved batch to {output_path}")
```

Bài tập 05C

1. # Đường dẫn dữ liệu và batch
2. data_test_path = '/content/drive/MyDrive/Bài tập CIFAR 10/dataset/uitTestCIFAR10'
3. output_dir = '/content/drive/MyDrive/Bài tập CIFAR 10/dataset_split/tiny'
4. # Tạo tên các lớp nhãn
5. classes = ['airplane', 'automobile', 'bird', 'cat', 'deer', 'dog', 'frog', 'horse', 'ship', 'truck']

Bài tập 05C

1. # Load dữ liệu uitTestCIFAR10
2. X_test, y_test = load_CIFAR_batch(data_test_path)
3. y_test = np.array(y_test)

4. # Lưu các ảnh ngẫu nhiên của mỗi lớp thành batch
5. save_class_images_to_batch_test(X_test, y_test, classes, output_dir, num_images_per_class=3)

Bài tập CIFAR 10

— Bài tập 06A:

- + Load bộ dữ liệu uitTrainCIFAR10 trong thư mục làm việc của Colab.
- + Tạo bộ dữ liệu uitSmallTrainCIFAR10 với mỗi lớp nhãn có 100 ảnh được lấy từ bộ dữ liệu uitTrainCIFAR10.

airplane



automobile



bird



cat



deer



dog



frog



horse



ship



truck



Bài tập 06A

– Hàm `load_CIFAR_batch`,
`save_class_images_to_batch_train` tương tự như bài tập 5A

Bài tập 06A

1. # Đường dẫn dữ liệu và batch
2. data_train_path = '/content/drive/MyDrive/Bài tập CIFAR 10/dataset/uitTrainCIFAR10'
3. output_dir = '/content/drive/MyDrive/Bài tập CIFAR 10/dataset_split/small'
4. # Tạo tên các lớp nhãn
5. classes = ['airplane', 'automobile', 'bird', 'cat', 'deer', 'dog', 'frog', 'horse', 'ship', 'truck']

Bài tập 06A

1. # Load dữ liệu uitTrainCIFAR10
2. X_train, y_train = load_CIFAR_batch(data_train_path)
3. y_train = np.array(y_train)

4. # Lưu các ảnh ngẫu nhiên của mỗi lớp thành batch
5. save_class_images_to_batch_train(X_train, y_train, classes, output_dir, num_images_per_class=100)

Bài tập CIFAR 10

— Bài tập 06B:

- + Load bộ dữ liệu uitValidationCIFAR10 trong thư mục làm việc của Colab.
- + Tạo bộ dữ liệu uitSmallValidationCIFAR10 với mỗi lớp nhãn có 25 ảnh được lấy từ bộ dữ liệu uitValidationCIFAR10.

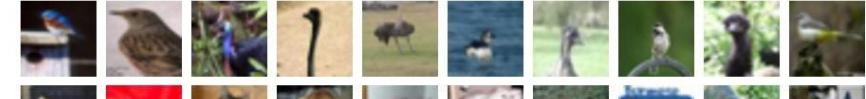
airplane



automobile



bird



cat



deer



dog



frog



horse



ship



truck

Bài tập 06B

– Hàm `load_CIFAR_batch`,
`save_class_images_to_batch_validation` tương tự như bài
tập 5B

Bài tập 06B

1. # Đường dẫn dữ liệu và batch
2. data_val_path = '/content/drive/MyDrive/Bài tập CIFAR 10/dataset/uitValidationCIFAR10'
3. output_dir = '/content/drive/MyDrive/Bài tập CIFAR 10/dataset_split/small'
4. # Tạo tên các lớp nhãn
5. classes = ['airplane', 'automobile', 'bird', 'cat', 'deer', 'dog', 'frog', 'horse', 'ship', 'truck']

Bài tập 06B

1. # Load dữ liệu uitValidationCIFAR10
2. X_val, y_val = load_CIFAR_batch(data_val_path)
3. y_val = np.array(y_val)

4. # Lưu các ảnh ngẫu nhiên của mỗi lớp thành batch
5. save_class_images_to_batch_validation(X_val, y_val, classes, output_dir, num_images_per_class=25)

Bài tập CIFAR 10

— Bài tập 06C:

- + Load bộ dữ liệu uitTestCIFAR10 trong thư mục làm việc của Colab.
- + Tạo bộ dữ liệu uitSmallTestCIFAR10 với mỗi lớp nhãn có 25 ảnh được lấy từ bộ dữ liệu uitTestCIFAR10.

airplane



automobile



bird



cat



deer



dog



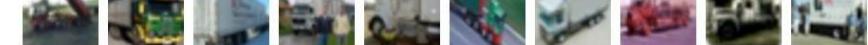
frog



horse



ship



truck

Bài tập 06C

- Hàm `load_CIFAR_batch`,
`save_class_images_to_batch_validation` tương tự như bài
tập 5C

Bài tập 06C

1. # Đường dẫn dữ liệu và batch
2. data_test_path = '/content/drive/MyDrive/Bài tập CIFAR 10/dataset/uitTestCIFAR10'
3. output_dir = '/content/drive/MyDrive/Bài tập CIFAR 10/dataset_split/small'
4. # Tạo tên các lớp nhãn
5. classes = ['airplane', 'automobile', 'bird', 'cat', 'deer', 'dog', 'frog', 'horse', 'ship', 'truck']

Bài tập 06C

1. # Load dữ liệu uitTestCIFAR10
2. X_test, y_test = load_CIFAR_batch(data_test_path)
3. y_test = np.array(y_test)

4. # Lưu các ảnh ngẫu nhiên của mỗi lớp thành batch
5. save_class_images_to_batch_test(X_test, y_test, classes, output_dir, num_images_per_class=25)

Bài tập CIFAR 10

— Bài tập 07A:

- + Load bộ dữ liệu uitTrainCIFAR10 trong thư mục làm việc của Colab.
- + Tạo bộ dữ liệu uitMediumTrainCIFAR10 với mỗi lớp nhãn có 1000 ảnh được lấy từ bộ dữ liệu uitTrainCIFAR10.

airplane



automobile



bird



cat



deer



dog



frog



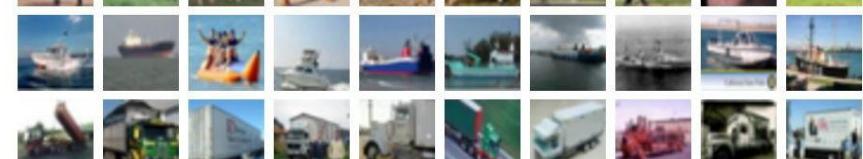
horse



ship



truck



Bài tập 07A

– Hàm `load_CIFAR_batch`,
`save_class_images_to_batch_train` tương tự như bài tập 5A

Bài tập 07A

1. # Đường dẫn dữ liệu và batch
2. data_train_path = '/content/drive/MyDrive/Bài tập CIFAR 10/dataset/uitTrainCIFAR10'
3. output_dir = '/content/drive/MyDrive/Bài tập CIFAR 10/dataset_split/medium'
4. # Tạo tên các lớp nhãn
5. classes = ['airplane', 'automobile', 'bird', 'cat', 'deer', 'dog', 'frog', 'horse', 'ship', 'truck']

Bài tập 07A

1. # Load dữ liệu uitTrainCIFAR10
2. X_train, y_train = load_CIFAR_batch(data_train_path)
3. y_train = np.array(y_train)

4. # Lưu các ảnh ngẫu nhiên của mỗi lớp thành batch
5. save_class_images_to_batch_train(X_train, y_train, classes, output_dir, num_images_per_class=1000)

Bài tập CIFAR 10

— Bài tập 07B:

- + Load bộ dữ liệu uitValidationCIFAR10 trong thư mục làm việc của Colab.
- + Tạo bộ dữ liệu uitMediumValidationCIFAR10 với mỗi lớp nhãn có 250 ảnh được lấy từ bộ dữ liệu uitValidationCIFAR10.

airplane



automobile



bird



cat



deer



dog



frog



horse



ship



truck



Bài tập 07B

– Hàm `load_CIFAR_batch`,
`save_class_images_to_batch_validation` tương tự như bài
tập 5B

Bài tập 07B

1. # Đường dẫn dữ liệu và batch
2. data_val_path = '/content/drive/MyDrive/Bài tập CIFAR 10/dataset/uitValidationCIFAR10'
3. output_dir = '/content/drive/MyDrive/Bài tập CIFAR 10/dataset_split/medium'
4. # Tạo tên các lớp nhãn
5. classes = ['airplane', 'automobile', 'bird', 'cat', 'deer', 'dog', 'frog', 'horse', 'ship', 'truck']

Bài tập 07B

1. # Load dữ liệu uitValidationCIFAR10
2. X_val, y_val = load_CIFAR_batch(data_val_path)
3. y_val = np.array(y_val)

4. # Lưu các ảnh ngẫu nhiên của mỗi lớp thành batch
5. save_class_images_to_batch_validation(X_val, y_val, classes, output_dir, num_images_per_class=250)

Bài tập CIFAR 10

— Bài tập 07C:

- + Load bộ dữ liệu uitTestCIFAR10 trong thư mục làm việc của Colab.
- + Tạo bộ dữ liệu uitMediumTestCIFAR10 với mỗi lớp nhãn có 250 ảnh được lấy từ bộ dữ liệu uitTestCIFAR10.

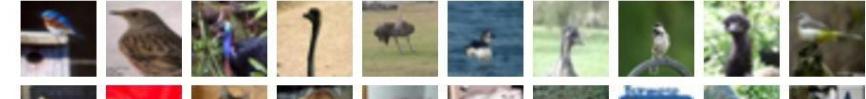
airplane



automobile



bird



cat



deer



dog



frog



horse



ship



truck



Bài tập 07C

- Hàm `load_CIFAR_batch`,
`save_class_images_to_batch_validation` tương tự như bài
tập 5C

Bài tập 07C

1. # Đường dẫn dữ liệu và batch
2. data_test_path = '/content/drive/MyDrive/Bài tập CIFAR 10/dataset/uitTestCIFAR10'
3. output_dir = '/content/drive/MyDrive/Bài tập CIFAR 10/dataset_split/medium'
4. # Tạo tên các lớp nhãn
5. classes = ['airplane', 'automobile', 'bird', 'cat', 'deer', 'dog', 'frog', 'horse', 'ship', 'truck']

Bài tập 07C

1. # Load dữ liệu uitTestCIFAR10
2. X_test, y_test = load_CIFAR_batch(data_test_path)
3. y_test = np.array(y_test)

4. # Lưu các ảnh ngẫu nhiên của mỗi lớp thành batch
5. save_class_images_to_batch_test(X_test, y_test, classes, output_dir, num_images_per_class=250)

Bài tập CIFAR 10

— Bài tập 08A:

- + Load bộ dữ liệu uitTrainCIFAR10 trong thư mục làm việc của Colab.
- + Tạo bộ dữ liệu uitHalfTrainCIFAR10 với mỗi lớp nhãn có 2000 ảnh được lấy từ bộ dữ liệu uitTrainCIFAR10.

airplane



automobile



bird



cat



deer



dog



frog



horse



ship



truck



Bài tập 08A

– Hàm `load_CIFAR_batch`,
`save_class_images_to_batch_train` tương tự như bài tập 5A

Bài tập 08A

1. # Đường dẫn dữ liệu và batch
2. data_train_path = '/content/drive/MyDrive/Bài tập CIFAR 10/dataset/uitTrainCIFAR10'
3. output_dir = '/content/drive/MyDrive/Bài tập CIFAR 10/dataset_split/half'
4. # Tạo tên các lớp nhãn
5. classes = ['airplane', 'automobile', 'bird', 'cat', 'deer', 'dog', 'frog', 'horse', 'ship', 'truck']

Bài tập 08A

1. # Load dữ liệu uitTrainCIFAR10
2. X_train, y_train = load_CIFAR_batch(data_train_path)
3. y_train = np.array(y_train)

4. # Lưu các ảnh ngẫu nhiên của mỗi lớp thành batch
5. save_class_images_to_batch_train(X_train, y_train, classes, output_dir, num_images_per_class=2000)

Bài tập CIFAR 10

— Bài tập 08B:

- + Load bộ dữ liệu uitValidationCIFAR10 trong thư mục làm việc của Colab.
- + Tạo bộ dữ liệu uitHalfValidationCIFAR10 với mỗi lớp nhãn có 500 ảnh được lấy từ bộ dữ liệu uitValidationCIFAR10.

airplane



automobile



bird



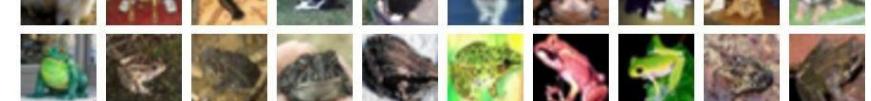
cat



deer



dog



frog



horse



ship



truck

Bài tập 08B

- Hàm `load_CIFAR_batch`,
`save_class_images_to_batch_validation` tương tự như bài
tập 5B

Bài tập 08B

1. # Đường dẫn dữ liệu và batch
2. data_val_path = '/content/drive/MyDrive/Bài tập CIFAR 10/dataset/uitValidationCIFAR10'
3. output_dir = '/content/drive/MyDrive/Bài tập CIFAR 10/dataset_split/half'
4. # Tạo tên các lớp nhãn
5. classes = ['airplane', 'automobile', 'bird', 'cat', 'deer', 'dog', 'frog', 'horse', 'ship', 'truck']

Bài tập 08B

1. # Load dữ liệu uitValidationCIFAR10
2. X_val, y_val = load_CIFAR_batch(data_val_path)
3. y_val = np.array(y_val)

4. # Lưu các ảnh ngẫu nhiên của mỗi lớp thành batch
5. save_class_images_to_batch_validation(X_val, y_val, classes, output_dir, num_images_per_class=500)

Bài tập CIFAR 10

— Bài tập 08C:

- + Load bộ dữ liệu uitTestCIFAR10 trong thư mục làm việc của Colab.
- + Tạo bộ dữ liệu uitHalfTestCIFAR10 với mỗi lớp nhãn có 500 ảnh được lấy từ bộ dữ liệu uitTestCIFAR10.

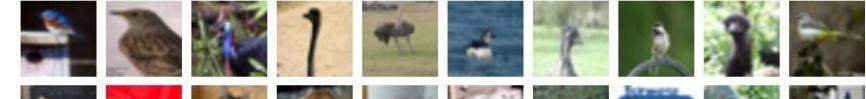
airplane



automobile



bird



cat



deer



dog



frog



horse



ship



truck



Bài tập 08C

- Hàm `load_CIFAR_batch`,
`save_class_images_to_batch_validation` tương tự như bài
tập 5C

Bài tập 08C

1. # Đường dẫn dữ liệu và batch
2. data_test_path = '/content/drive/MyDrive/Bài tập CIFAR 10/dataset/uitTestCIFAR10'
3. output_dir = '/content/drive/MyDrive/Bài tập CIFAR 10/dataset_split/half'
4. # Tạo tên các lớp nhãn
5. classes = ['airplane', 'automobile', 'bird', 'cat', 'deer', 'dog', 'frog', 'horse', 'ship', 'truck']

Bài tập 08C

1. # Load dữ liệu uitTestCIFAR10
2. X_test, y_test = load_CIFAR_batch(data_test_path)
3. y_test = np.array(y_test)

4. # Lưu các ảnh ngẫu nhiên của mỗi lớp thành batch
5. save_class_images_to_batch_test(X_test, y_test, classes, output_dir, num_images_per_class=500)

BÀI TẬP KNN

Bài tập *kNN*

- Bài tập 11:
 - + Load bộ dữ liệu uitTinyTrainCIFAR10 trong thư mục làm việc của Colab.
 - + Xây dựng mô hình bằng thuật toán *kNN* với $k = 1$ và sử dụng độ đo khoảng cách Manhattan.
 - + Tính độ chính xác của mô hình kết quả trên bộ dữ liệu uitTinyTrainCIFAR10 , uitTinyValidationCIFAR10 và uitTinyTestCIFAR10.

Bài tập 11

```
11.# Định nghĩa hàm để load batch dữ liệu
12.def load_batch(batch_name):
13.    import pickle
14.    with open(batch_name, 'rb') as f:
15.        batch = pickle.load(f, encoding='bytes')
16.        X = batch[b'data']
17.        Y = batch[b'labels']
18.    return X, Y
```

Bài tập 11

11.# Đường dẫn đến thư mục chứa bộ dữ liệu

12.folder_path = '/content/drive/MyDrive/Bài tập CIFAR
10/dataset_split/tiny'

13.# Tạo tên các lớp nhãn

14.labels = ['Airplane', 'Automobile', 'Bird', 'Cat',
'Deer', 'Dog', 'Frog', 'Horse', 'Ship', 'Truck']

15....

Bài tập 11

11....

12.# Load dữ liệu uitTinyTrainCIFAR10

13.batch_name = os.path.join(folder_path,
'uitTinyTrainCIFAR10')

14.X_train, y_train = load_batch(batch_name)

15.y_train = np.array(y_train)

16....

Bài tập 11

11....

12.# Load dữ liệu uitTinyValidationCIFAR10

13.batch_name = os.path.join(folder_path,
'uitTinyValidationCIFAR10')

14.X_val, y_val = load_batch(batch_name)

15.y_val = np.array(y_val)

16....

Bài tập 11

11....

12.# Load dữ liệu uitTinyTestCIFAR10

13.batch_name = os.path.join(folder_path,
'uitTinyTestCIFAR10')

14.X_test, y_test = load_batch(batch_name)

15.y_test = np.array(y_test)

Bài tập 11

```
11. class kNearestNeighbor:  
12.     def __init__(self):  
13.         pass  
14.     ...
```

Bài tập 11

```
11.class kNearestNeighbor:  
12.    def train(self, X, y):  
13.        """  
14.            X is N x D where each row is an example. Y is 1-  
15.            dimension of size N  
16.            """  
17.            # the nearest neighbor classifier simply remembers  
18.            # all the training data  
19.            self.X_train = X  
20.            self.y_train = y  
21.  
22.            ...
```

Bài tập 11

```
11.class kNearestNeighbor:  
12.    def predict(self, X, k=1):  
13.        """  
14.            X is N x D where each row is an example we wish to  
15.            predict label for  
16.            """  
17.            num_test = X.shape[0]  
18.            # let's make sure that the output type matches the  
19.            # input type  
20.            Y_pred = np.zeros(num_test, dtype=self.y_train.dtype)  
21.  
22.            ...  
23.            # loop over all test rows
```

Bài tập 11

```
11. class kNearestNeighbor:  
12.     def predict(self, X, k=1):  
13.         ...  
14.         # loop over all test rows  
15.         for i in range(num_test):  
16.             # find the k nearest training image to the i'th test image  
17.             # using the L1 distance (sum of absolute value differences)  
18.             distances = np.sum(np.abs(self.X_train - X[i, :]), axis=1)  
19.             min_indices = np.argsort(distances)[:k]  
20.             closest_y = self.y_train[min_indices]  
21.             Y_pred[i] = np.argmax(np.bincount(closest_y))  
22.     return Y_pred
```

Bài tập 11

11.#Traing model

12.classifier = kNearestNeighbor()

13.classifier.train(X_train, y_train)

Bài tập 11

```
1.y_train_pred = classifier.predict(X_train)  
2.y_val_pred = classifier.predict(X_val)  
3.y_test_pred = classifier.predict(X_test)
```

Bài tập 11

```
11.# Compute and display the accuracy in train set
12.num_training = y_train.shape[0]
13.num_correct = np.sum(y_train_pred == y_train)
14.accuracy = float(num_correct) / num_training
15.print('Got %d / %d correct => accuracy: %f' %
        (num_correct, num_training, accuracy))
```

Bài tập 11

```
11.# Compute and display the accuracy in validation set
12.num_val = y_val.shape[0]
13.num_correct = np.sum(y_val_pred == y_val)
14.accuracy = float(num_correct) / num_val
15.print('Got %d / %d correct => accuracy: %f' %
        (num_correct, num_val, accuracy))
```

Bài tập 11

```
11.# Compute and display the accuracy in test set
12.num_test = y_test.shape[0]
13.num_correct = np.sum(y_test_pred == y_test)
14.accuracy = float(num_correct) / num_test
15.print('Got %d / %d correct => accuracy: %f' %
        (num_correct, num_test, accuracy))
```

Bài tập 12

— Bài tập 12:

- + Load bộ dữ liệu uitTinyTrainCIFAR10 trong thư mục làm việc của Colab.
- + Xây dựng mô hình bằng thuật toán kNN với $k = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20$ và sử dụng độ đo khoảng cách Manhattan.
- + Chỉ ra siêu tham số k tốt nhất.

Bài tập 12

11. # Tìm siêu tham số k tốt nhất

12. list_k = [1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20]

13. best_k = 0

14. best_accuracy = 0

15. for i in list_k:

16. y_val_pred = classifier.predict(X_val, k = i)

17. accuracy = calc_accuracy(y_val, y_val_pred)

18. if accuracy > best_accuracy:

19. best_k = i

20. best_accuracy = accuracy

21. print(f"when k = {i}, accuracy is {round(accuracy, 4)} | best:
{round(best_accuracy, 4)} (k={best_k}) ")

Bài tập 13

— Bài tập 13:

- + Load bộ dữ liệu uitTinyTrainCIFAR10 trong thư mục làm việc của Colab.
- + Xây dựng mô hình bằng thuật toán kNN với $k = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20$ và sử dụng độ đo khoảng cách Euclidean.
- + Chỉ ra siêu tham số k tốt nhất.

Bài tập 13

```
11. class kNearestNeighbor:  
12.     def __init__(self):  
13.         pass  
14.     ...
```

Bài tập 13

```
11.class kNearestNeighbor:  
12.    def train(self, x, y):  
13.        # the nearest neighbor classifier simply  
             remembers all the training data  
14.        self.x_train = x  
15.        self.y_train = y  
16.        ...
```

Bài tập 13

```
11.class kNearestNeighbor:  
12.    def predict(self, X, k=1):  
13.        num_test = X.shape[0]  
14.        Y_pred = np.zeros(num_test,  
15.                           dtype=self.y_train.dtype)  
16.        distances = np.sqrt(np.sum((self.X_train - X[i, :])**2,  
17.                               axis=1))  
18.        min_indices = np.argsort(distances)[:k]  
19.        closest_y = self.y_train[min_indices]  
20.        Y_pred[i] = np.argmax(np.bincount(closest_y))  
  
21.    return Y_pred
```

Bài tập 14

— Bài tập 14:

- + Load bộ dữ liệu uitTinyTrainCIFAR10 trong thư mục làm việc của Colab.
- + Xây dựng mô hình bằng thuật toán kNN với $k = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20$ và sử dụng độ đo khoảng cách Euclidean, Manhattan.
- + Chỉ ra siêu tham số độ đo khoảng cách và k tốt nhất.

Bài tập 14

```
11. class kNearestNeighbor:  
12.     def __init__(self):  
13.         pass  
14.     ...
```

Bài tập 14

```
11. class kNearestNeighbor:  
12.     def train(self, X, y):  
13.         # the nearest neighbor classifier simply  
             remembers all the training data  
14.         self.X_train = X  
15.         self.y_train = y  
16.     ...
```

Bài tập 14

```
11. class kNearestNeighbor:  
12.     def predict(self, X, k=1,  
13.                 distance_metric='manhattan'):  
14.         num_test = X.shape[0]  
15.         Y_pred = np.zeros(num_test, dtype=self.y_train.dtype)  
  
16.         for i in range(num_test):  
17.             if distance_metric == 'manhattan':  
18.                 distances = np.sum(np.abs(self.X_train - X[i, :]), axis=1)  
19.             elif distance_metric == 'euclidean':  
20.                 distances = np.sqrt(np.sum((self.X_train - X[i, :])**2, axis=1))  
21.             else:  
22.                 raise ValueError("Unknown distance metric")  
23.             ...  
24.             min_indices = np.argsort(distances)[:k]  
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN, ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH  
Closest_y = self.y_train[min_indices]
```

Bài tập 14

```
11. class kNearestNeighbor:  
12.     def predict(self, X, k=1,  
                     distance_metric='manhattan'):  
13.         ...  
14.         min_indices = np.argsort(distances)[:k]  
15.         closest_y = self.y_train[min_indices]  
16.         Y_pred[i] = np.argmax(np.bincount(closest_y))  
  
17.     return Y_pred
```

Bài tập 14

```
11.metrics = ['manhattan', 'euclidean']
12.best_metric = ''
13.list_k = [1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20]
14.best_k = 0
15.best_accuracy = 0
16....
```

Bài tập 14

```
11....  
12.for metric in metrics:  
13.    for i in list_k:  
14.        y_val_pred = classifier.predict(X_val, k = i, distance_metric =  
                                         metric)  
15.        accuracy = calc_accuracy(y_val, y_val_pred)  
16.        if accuracy > best_accuracy:  
17.            best_k = i  
18.            best_metric = metric  
19.            best_accuracy = accuracy  
20.        print(f"when k = {i} and metric = {metric}, accuracy is  
             {round(accuracy, 4)} | best: {round(best_accuracy, 4)} (k={best_k},  
                                         metric={best_metric}) ")
```

Bài tập *kNN*

— Bài tập 15:

- + Load bộ dữ liệu uitSmallTrainCIFAR10 trong thư mục làm việc của Colab.
- + Xây dựng mô hình bằng thuật toán *kNN* với $k = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20$ và sử dụng độ đo khoảng cách Euclidean, Manhattan.
- + Chỉ ra siêu tham số độ đo khoảng cách và k tốt nhất.

Bài tập *kNN*

— Bài tập 16:

- + Load bộ dữ liệu uitMediumTrainCIFAR10 trong thư mục làm việc của Colab.
- + Xây dựng mô hình bằng thuật toán *kNN* với $k = 1, 3, 5, 7, 9$ và sử dụng độ đo khoảng cách Euclidean, Manhattan.
- + Chỉ ra siêu tham số độ đo khoảng cách và k tốt nhất.

BÀI TẬP LINEAR CLASSIFIER

Bài tập linear classifier

— Bài tập 21:

- + Load bộ dữ liệu uitTinyTrainCIFAR10 trong thư mục làm việc của Colab.
- + Phát sinh ngẫu nhiên ma trận W .
- + Tính điểm cho từng điểm dữ liệu.
- + Tính độ chính xác.