CS231n Lecture 2

- BOAZ 10기 박성현
- BOAZ 11기 김태희
- BOAZ 11기 홍지민

01 Image Classification

Image Classification: A core task in Computer Vision

This image by Nikita is licensed under CC-BY 2.0

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

→ cat

Image Classification

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
```

Example training set

airplane

automobile

bird

cat

deer

01 Image Classification

First classifier: **Nearest Neighbor**

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

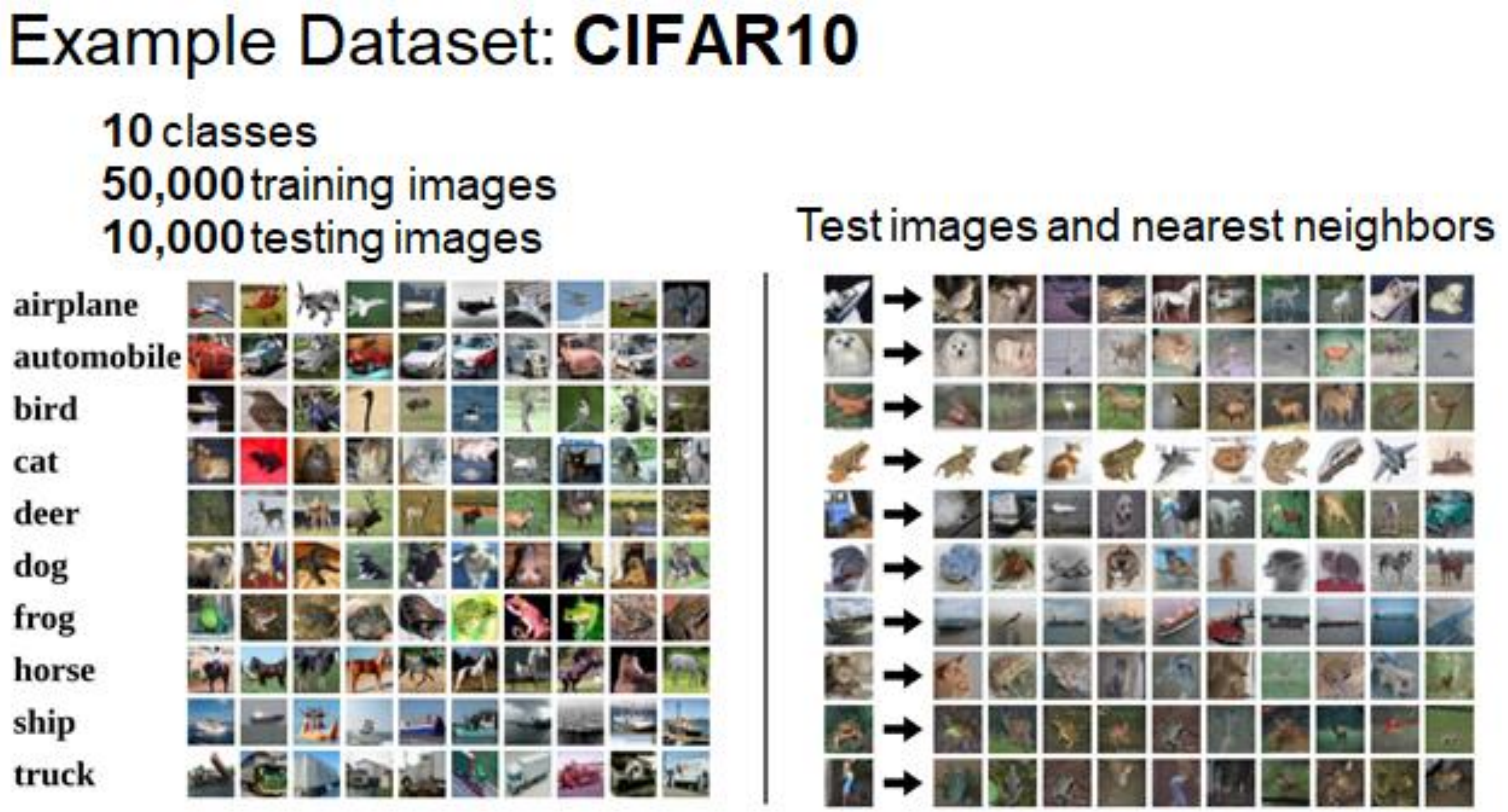
→ Memorize all data and labels

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

→ Predict the label of the most similar training image

01

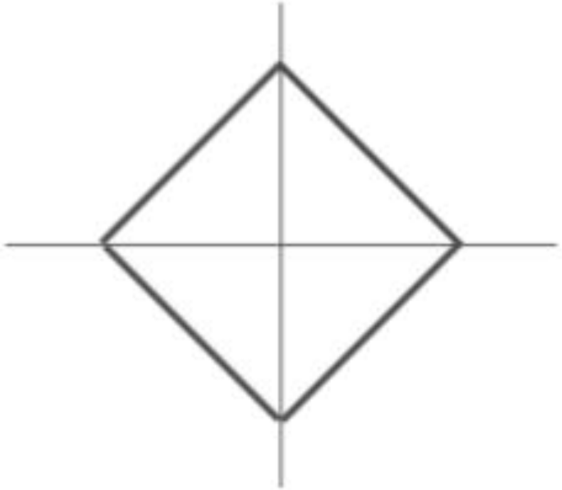
Image Classification



Distance Metric

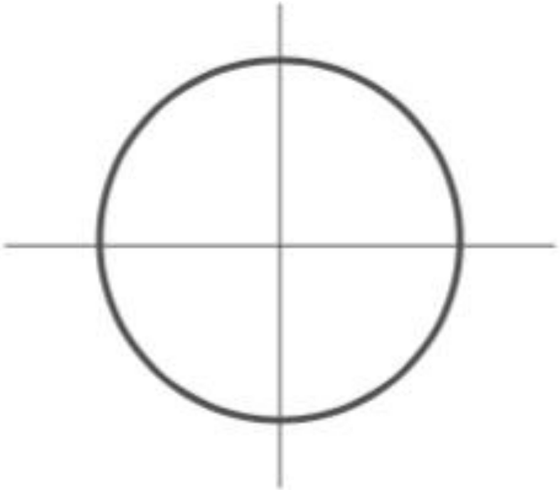
L1 (Manhattan) distance

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



L2 (Euclidean) distance

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



Distance Metric to compare images

L1 distance: $d_1(I_1, I_2) = \sum_p |I_1^p - I_2^p|$

test image

56	32	10	18
90	23	128	133
24	26	178	200
2	0	255	220

-

training image

10	20	24	17
8	10	89	100
12	16	178	170
4	32	233	112

=

pixel-wise absolute value differences

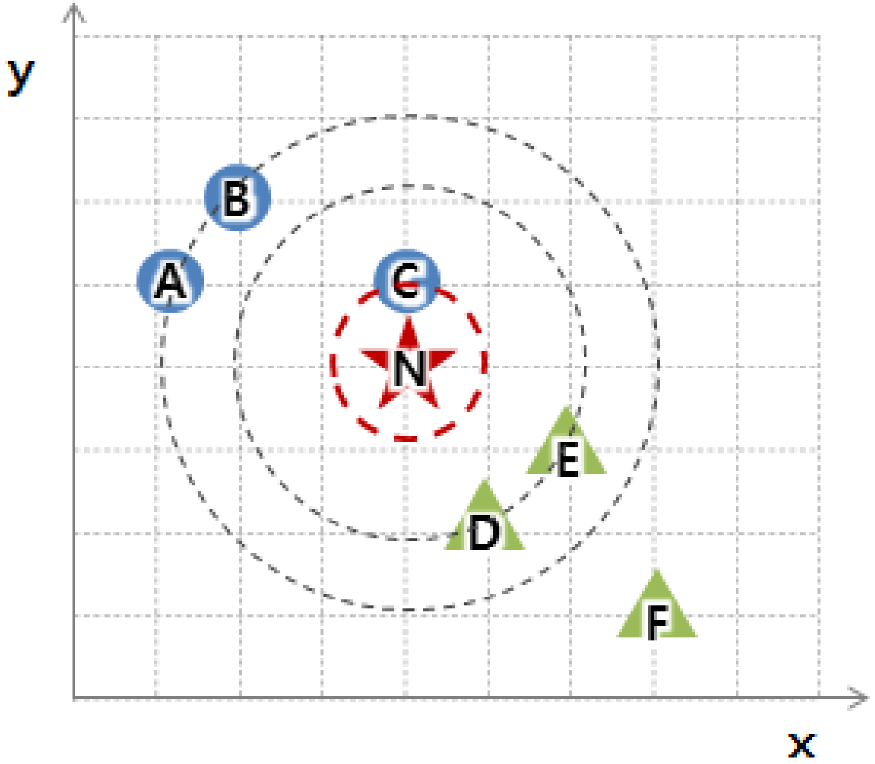
46	12	14	1
82	13	39	33
12	10	0	30
2	32	22	108

$\xrightarrow{\text{add}}$ 456

KNN (K-Nearest Neighbors)

K-Nearest Neighbor은 가까이 있는 k개 데이터로 데이터를 분류하는 알고리즘.

Q. 그림에서 K=1, K=3, K=5일 때, N은 어떤 데이터로 분류되는가?



KNN (K-Nearest Neighbors)

L1 (Manhattan) distance

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

K = 1

L2 (Euclidean) distance

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

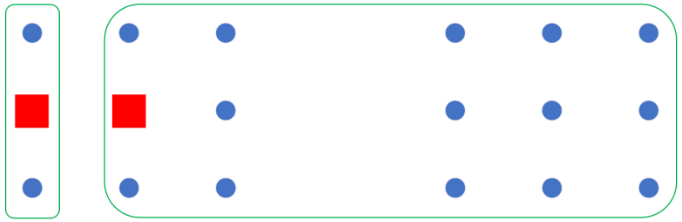
K = 1

Distance를 구하는 방법에 따라 K-NN의 결과가 달라질 수 있음.

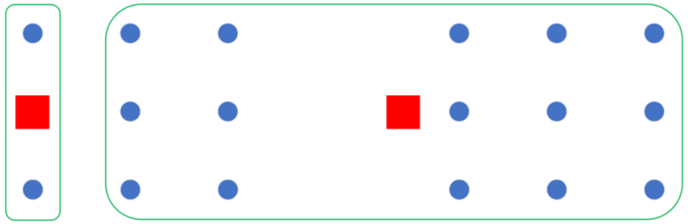
KNN (K-Nearest Neighbors)

(추가) K-Means Clustering

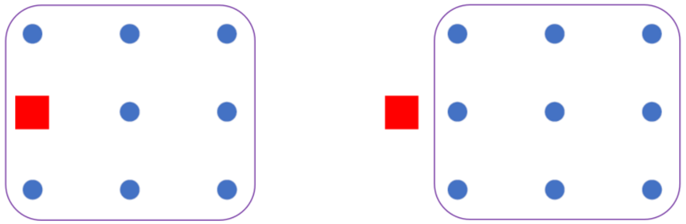
EM 알고리즘 기반으로 작동 (Expectation 스텝 & Maximization 스텝) – K는 중심의 개수



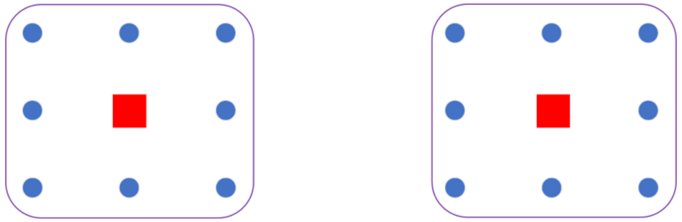
[1] 모든 파란 점을 가까운 중심에
Clustering (Expectation 스텝)



[2] 중심을 Cluster(군집)에 맞게
업데이트 (Maximization 스텝)



[3] 모든 파란 점을 가까운 중심에
Clustering (Expectation 스텝)



[4] 이 과정을 중심의 위치가
수렴할 때까지 반복

참고 : https://ratsgo.github.io/machine%20learning/2017/04/19/KC/

KNN (K-Nearest Neighbors)

유사하게 보이는 사진끼리 Distance가 가까움!!
Image Classification은 제대로 되지 않고 있는 모습!!!

Hyperparameter

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.

Hyperparameter란 학습을 할 때에 더 효과가 좋도록 하는 주 변수가 아닌 자동 설정 되는 변수를 의미한다. (학습자가 직접 설정해주는 변수!!!)
(Ex. Learning rate, batch size, epoch, hidden layer의 개수 등)

Hyperparameter

Setting Hyperparameters

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

BAD: K = 1 always works perfectly on training data

Your Dataset

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

BAD: No idea how algorithm will perform on new data

train	test

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

Better!

train	validation	test

Train / Test 비율은 일반적으로 70/30
데이터가 많은 경우에는 90/10으로 사용

새로운 방법 참고 : https://www.youtube.com/watch?v=AK60jzjDvlg&t=421s

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

04 KNN이 사용되지 않는 이유

k-Nearest Neighbor on images **never used.**

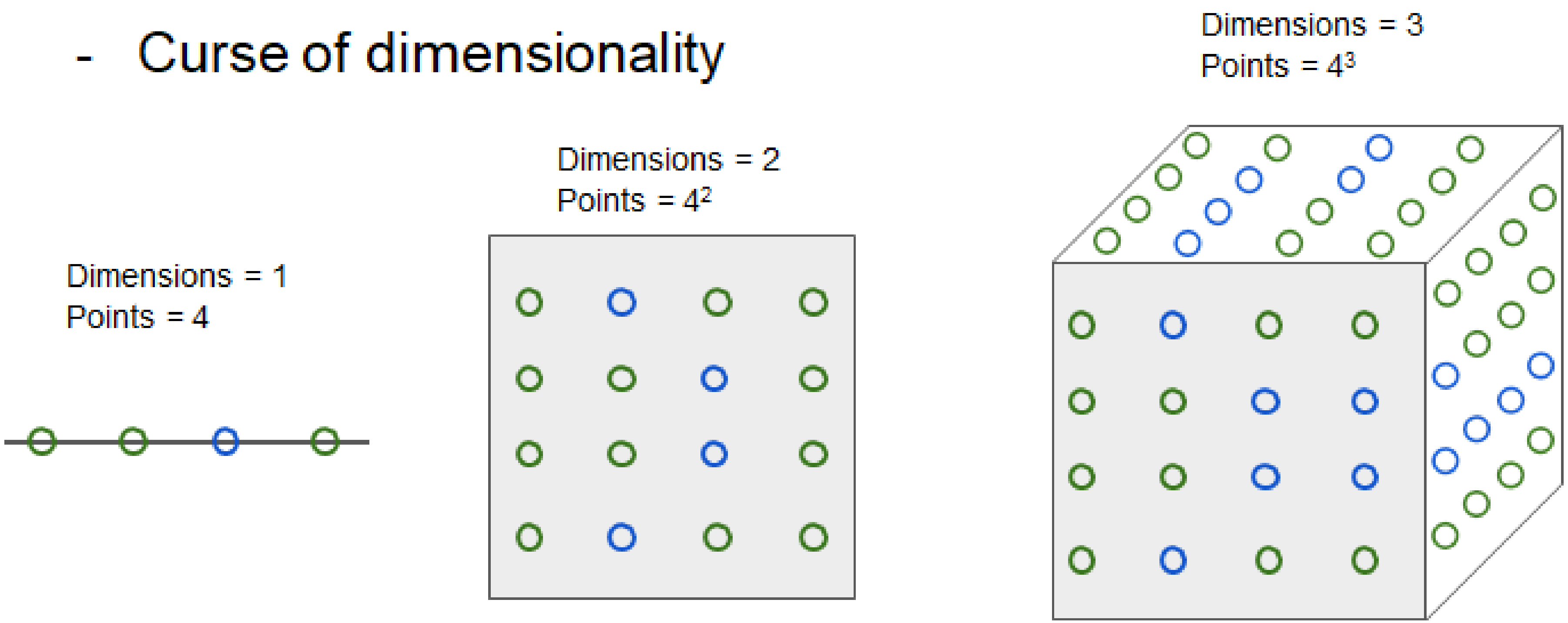
- Very slow at test time
- Distance metrics on pixels are not informative

(all 3 images have same L2 distance to the one on the left)

KNN이 사용되지 않는 이유

k-Nearest Neighbor on images **never used.**

- Curse of dimensionality

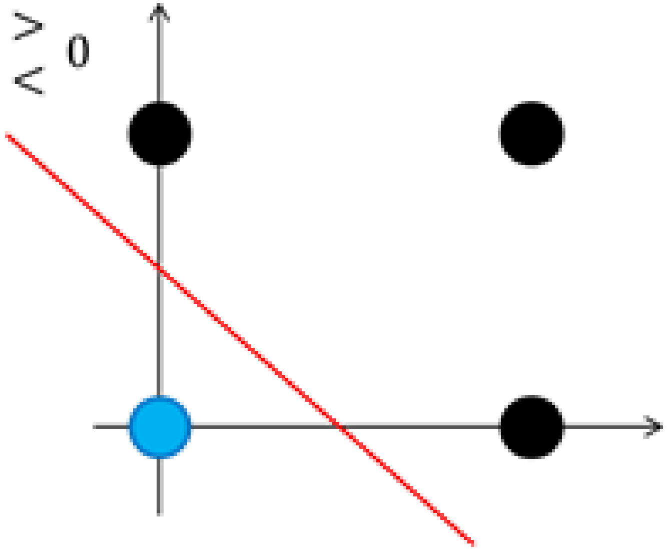


Linear Classification

Linear Classifier의 쉬운 예시

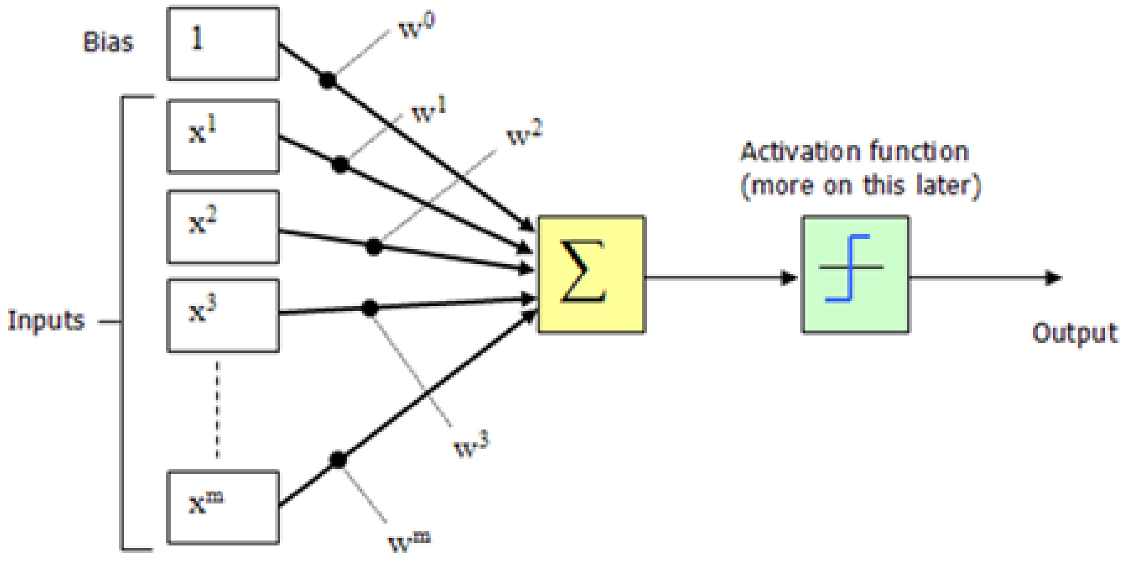
$$d(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b > 0 \text{ 이면 } \rightarrow \mathbf{x} \in \omega_1$$
$$d(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b < 0 \text{ 이면 } \rightarrow \mathbf{x} \in \omega_2$$

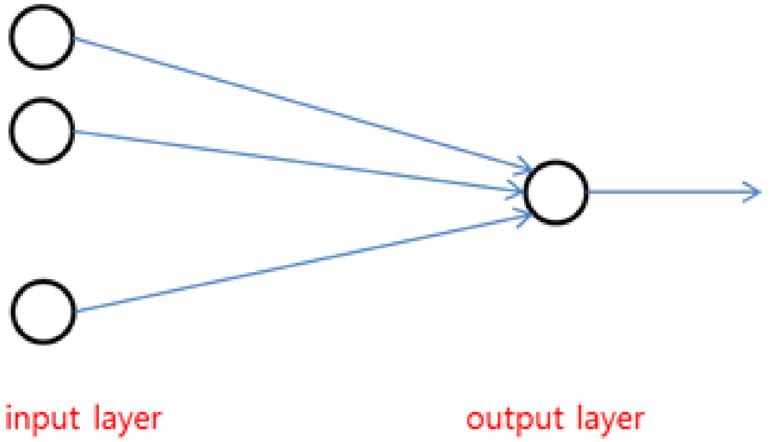
$$d(\mathbf{x}) = x_1 + x_2 - 0.5 \underset{<}{>} 0$$



04 Linear Classification

Linear Classification은 Neural Network에 사용된다.
(Activation Function에 대해서는 이후 강의에 나오지만,
Logistic Regression은 먼저 공부해보는 것을 추천)





04 Linear Classification

3072x1

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

10x1

10x1 10x3072

Image

Array of **32x32x3** numbers
(3072 numbers total)

$\longrightarrow f(x, W) \longrightarrow$ **10** numbers giving class scores

$\uparrow$

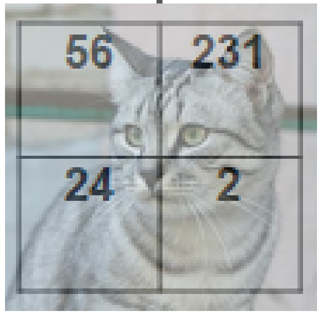
W

parameters
or weights

04 Linear Classification

Example with an image with 4 pixels, and 3 classes (cat/dog/ship)

Stretch pixels into column

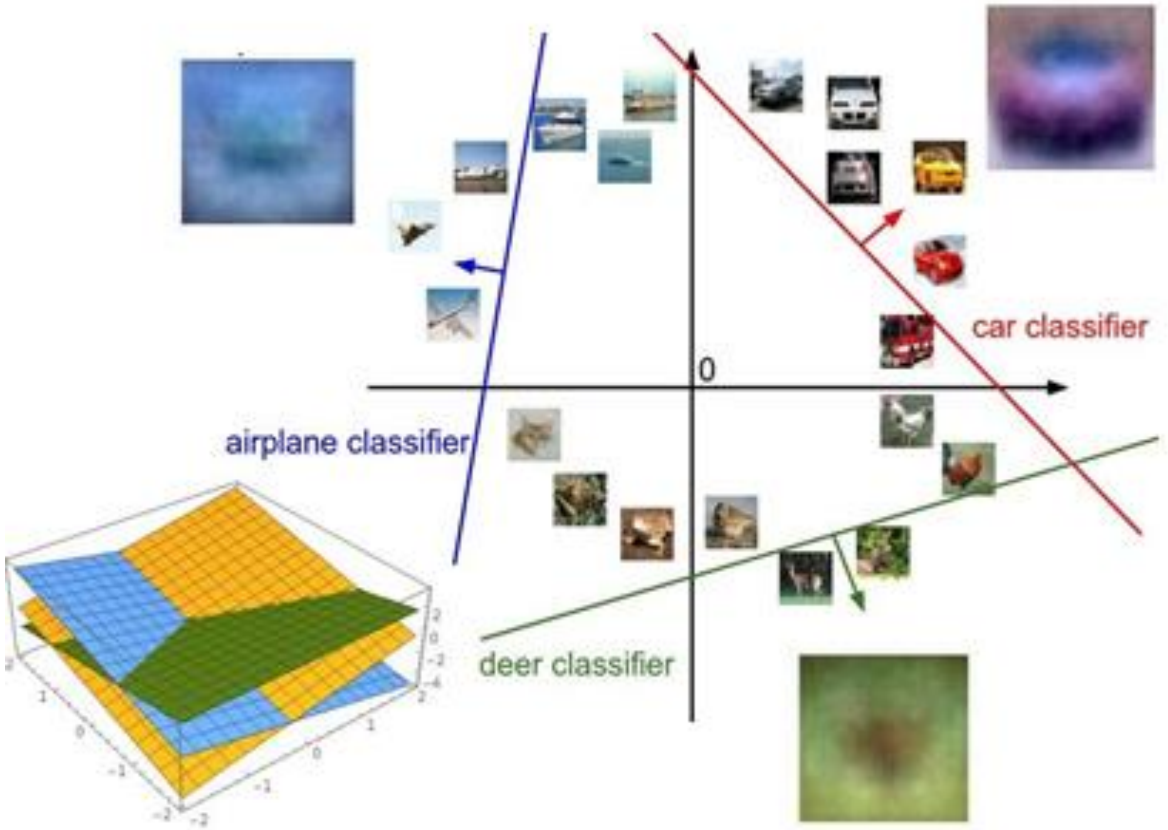


Input image

$$\underbrace{\begin{bmatrix} 0.2 & -0.5 & 0.1 & 2.0 \\ 1.5 & 1.3 & 2.1 & 0.0 \\ 0 & 0.25 & 0.2 & -0.3 \end{bmatrix}}_{W} \begin{bmatrix} 56 \\ 231 \\ 24 \\ 2 \end{bmatrix} + \begin{bmatrix} 1.1 \\ 3.2 \\ -1.2 \end{bmatrix} = \begin{bmatrix} -96.8 \\ 437.9 \\ 61.95 \end{bmatrix} \begin{array}{l} \text{Cat score} \\ \text{Dog score} \\ \text{Ship score} \end{array}$$

Linear Classification

Interpreting a Linear Classifier



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

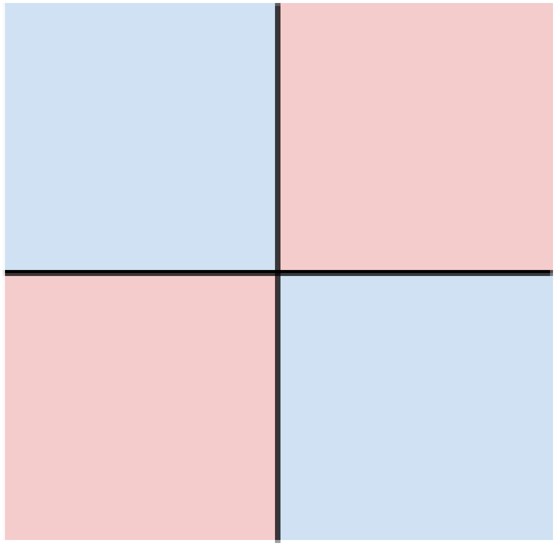
Array of **32x32x3** numbers
(3072 numbers total)

04 Linear Classification

Hard cases for a linear classifier

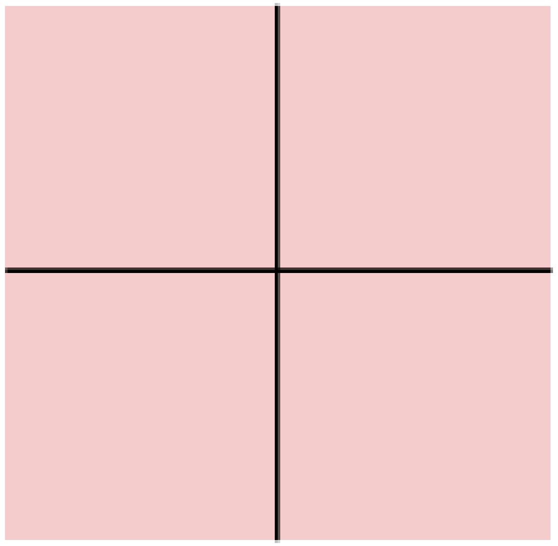
Class 1:
number of pixels > 0 odd

Class 2:
number of pixels > 0 even



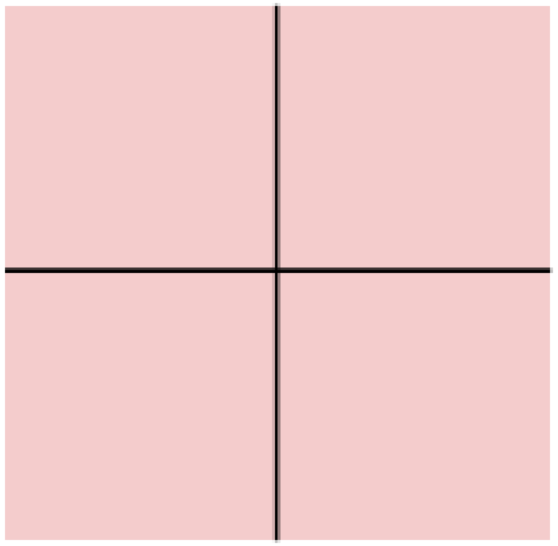
Class 1:
1 <= L2 norm <= 2

Class 2: Every thing else



Class 1: Three modes

Class 2: Every thing else



So far: Defined a (linear) score function $f(x,W) = Wx + b$

Example class scores for 3 images for some W:

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



airplane	-3.45	-0.51	3.42
automobile	-8.87	6.04	4.64
bird	0.09	5.31	2.65
cat	2.9	-4.22	5.1
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	-1.5
ship	-0.36	-2.09	-4.79
truck	-0.72	-2.93	6.14

End

참고 자료

CS231n : http://cs231n.stanford.edu/syllabus.html

K-NN 참고 : http://kkokkilkon.tistory.com/14

Ratsgo's Blog : https://ratsgo.github.io/machine%20learning/2017/04/19/KC/

Linear Classifier 참고 : http://www.whydsp.org/237