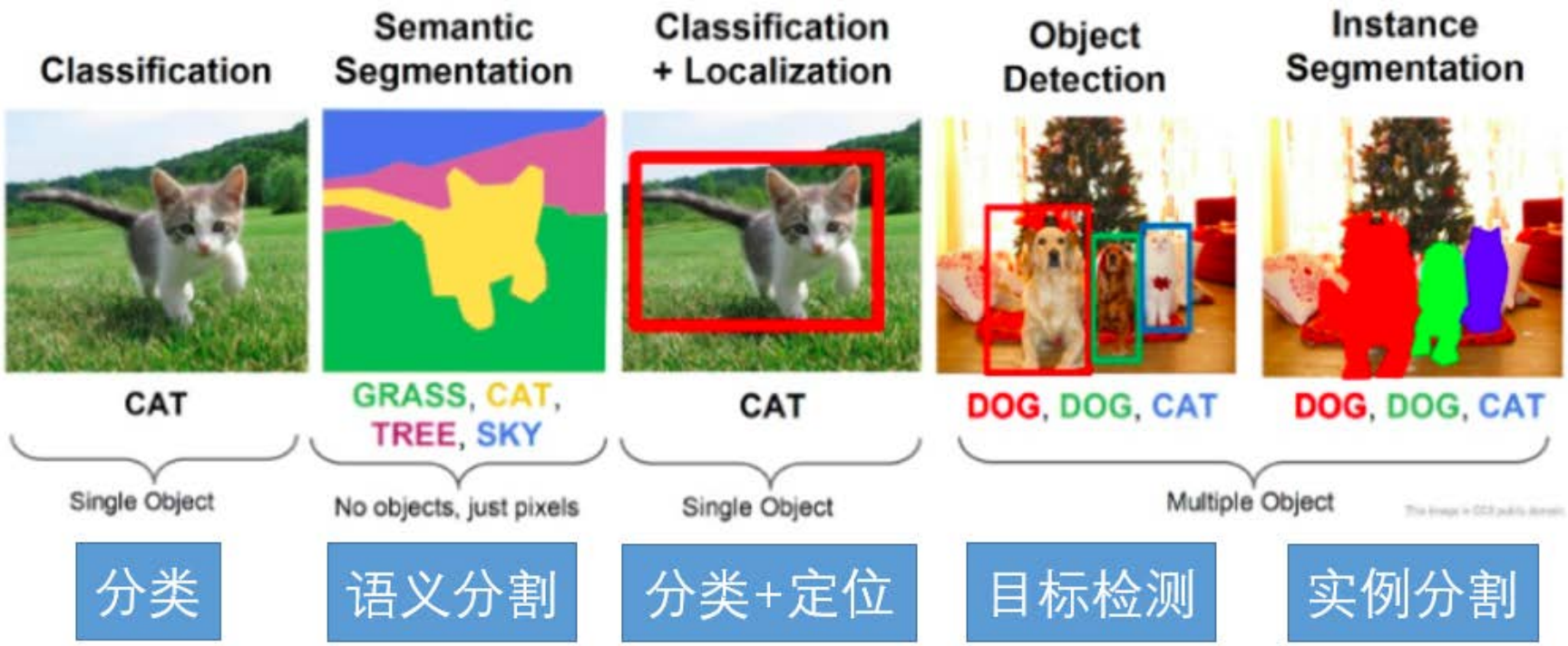
计算机视觉
Computer Vision

Lecture 2: 图像分类

上节课回顾

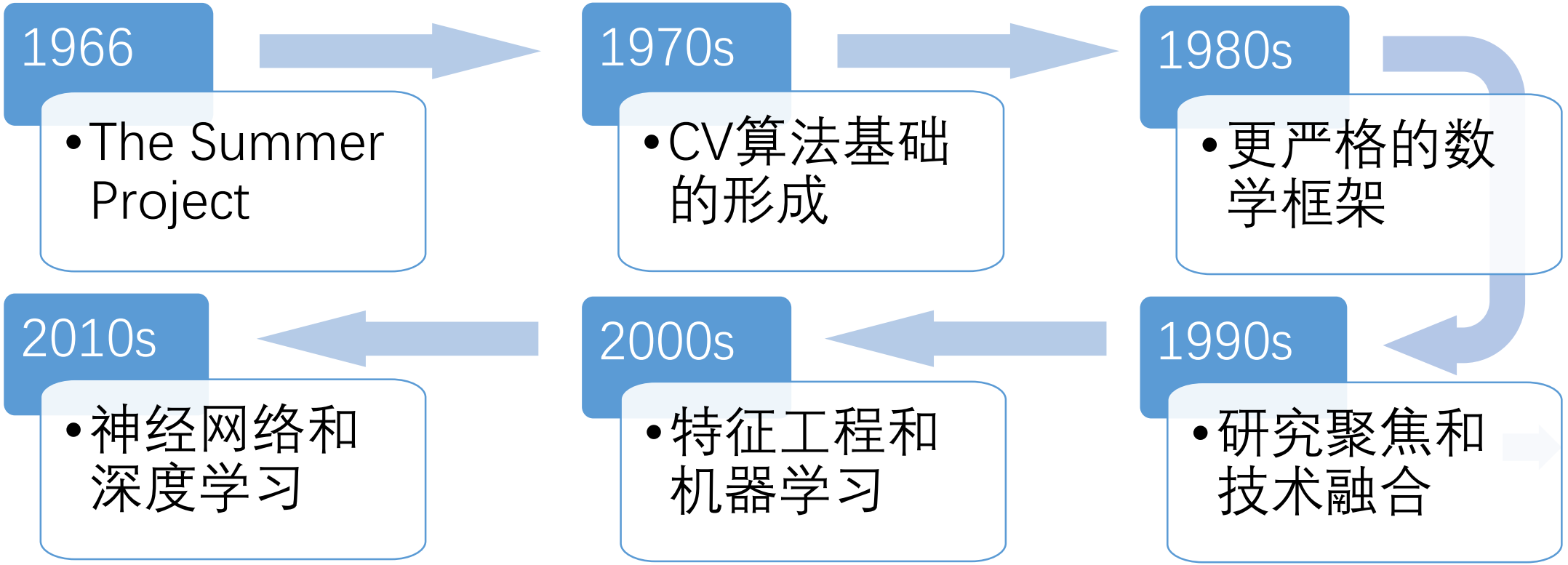
计算机视觉的定义和基本任务

- 用计算机模拟人类的视觉系统，去完成人类的视觉任务



上节课回顾

计算机视觉的发展历史



上节课回顾

本课程主要内容

CV基础

- 图像分类
- 损失函数和优化
- 神经网络

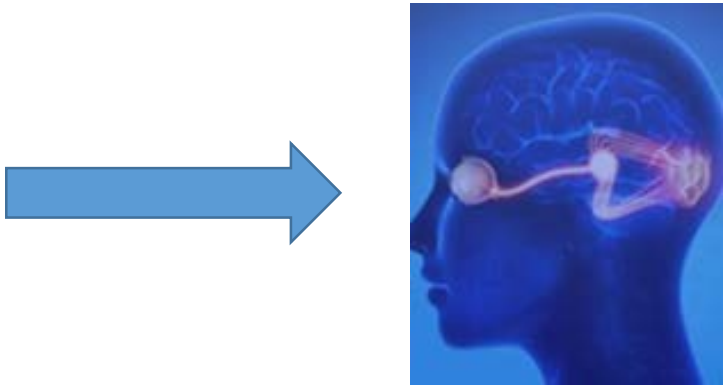
神经网络与CV

- 深度学习软件
- NN的训练
- CNN的结构演化

CV进阶任务

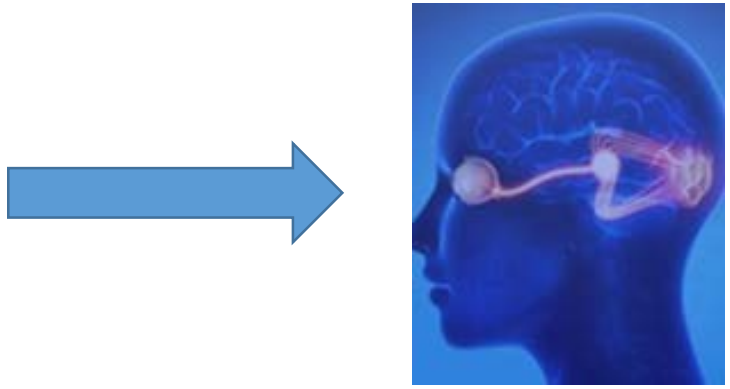
- 定位、检测、分割
- 生成模型
- 风格迁移

图像分类：CV的核心任务

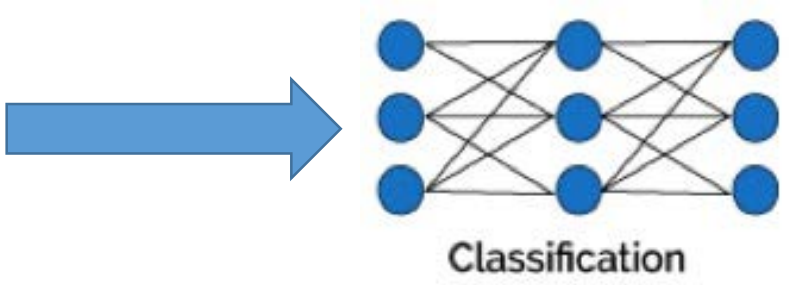


狗

图像分类：CV的核心任务



狗



？

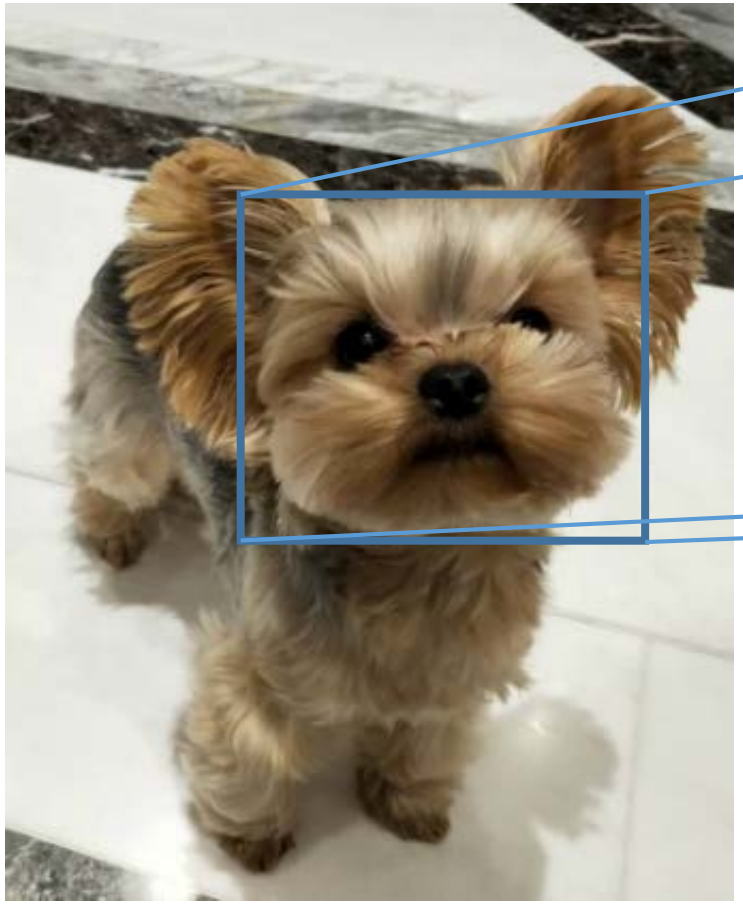
狗

猫

狼

（预设的一组类别）

对计算机的挑战：语义鸿沟



255	255	255	245	200	186	187	192	222	255	255	255
255	255	239	162	154	174	179	172	155	197	227	236
255	255	173	159	206	196	194	200	184	155	241	253
255	242	167	193	192	193	192	205	172	150	247	255
250	255	189	157	196	207	199	167	141	186	219	248
251	255	245	179	153	138	144	190	241	255	227	175
247	207	214	233	198	190	235	255	255	255	219	160
225	207	245	255	255	254	210	181	177	231	220	146
215	255	252	255	245	165	154	187	179	156	230	154
82	255	255	253	179	169	205	192	205	156	197	253
39	255	255	224	161	198	192	192	204	154	192	255
104	255	255	206	159	203	192	197	190	152	235	255
255	255	255	227	142	189	202	186	145	206	255	255
255	249	251	254	200	153	167	171	215	254	252	255
252	253	255	255	255	233	221	245	255	250	250	255
255	255	172	175	255	255	255	253	249	249	253	255

一组[0, 255]的数字 → 狗

Semantic Gap

对计算机的挑战：尺度差异

Scale Variation

对计算机的挑战：视角差异

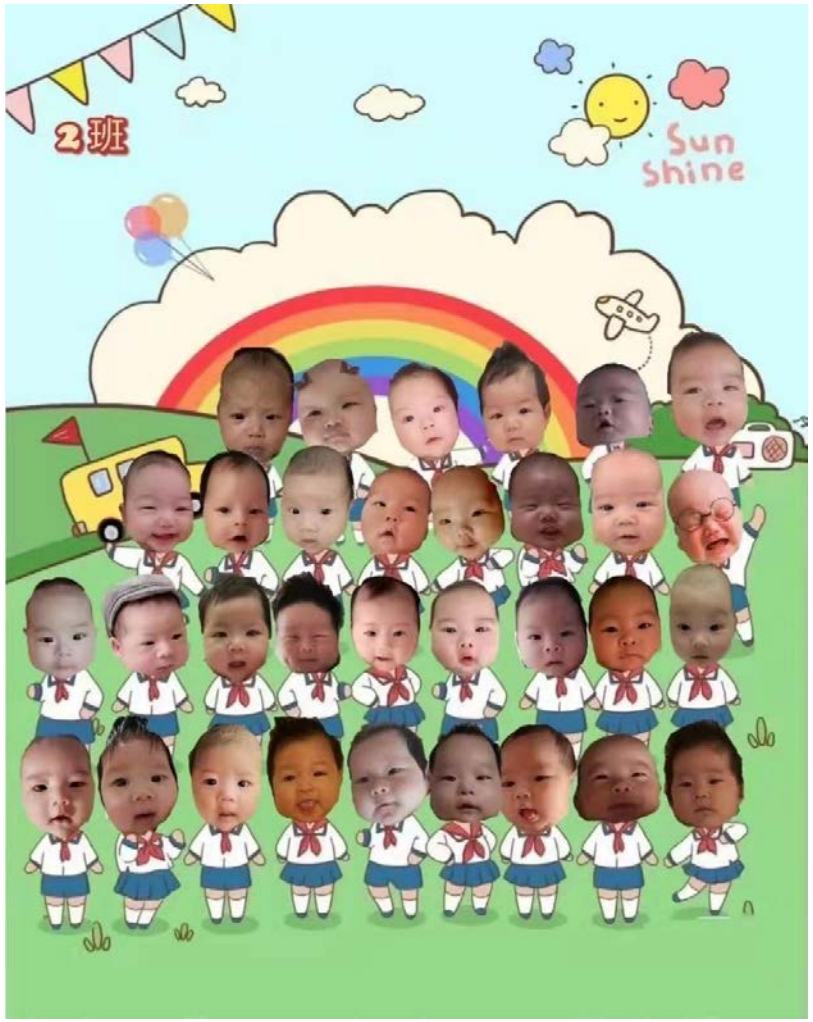



Viewpoint Variation

对计算机的挑战：类内差异

Intraclass Variation

对计算机的挑战：类内差异



Intraclass Variation

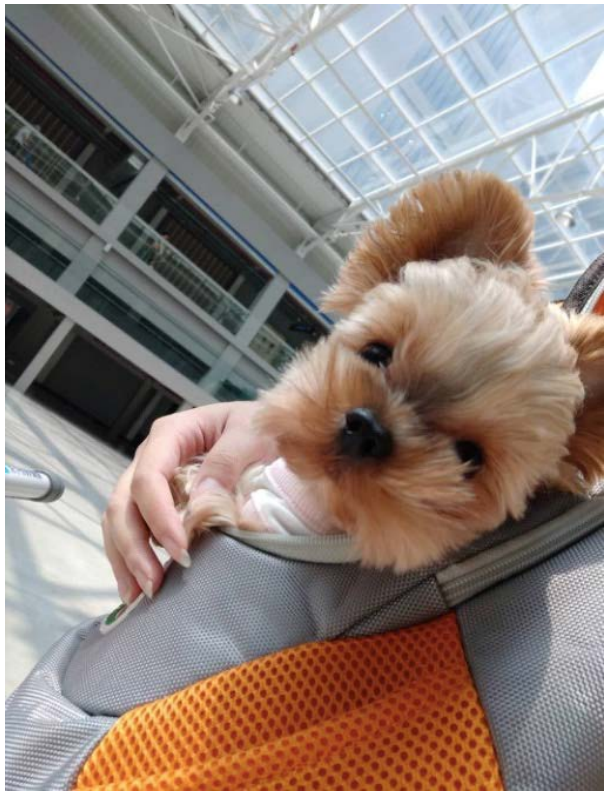
对计算机的挑战：其他

照明

illumination

背景混淆

background clutter



遮挡

occlusion

变形

deformation

早期的分类方法：设计特定的算法

寻找边和线

寻找角落和轮廓

眼

耳

鼻

……

毛

狗

早期的分类方法：设计特定的算法

眼

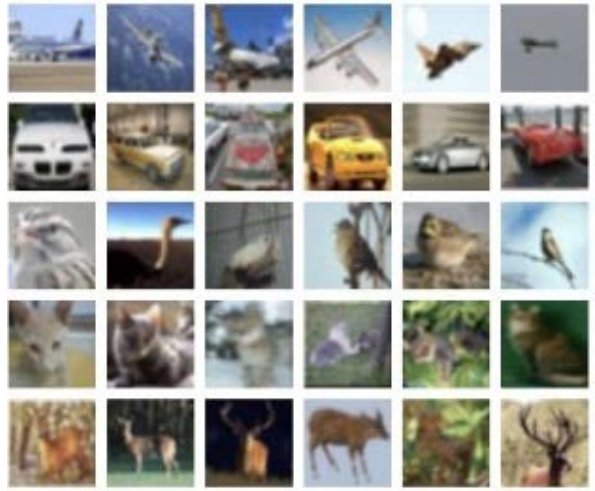
耳

1）健壮性差，对于不同的光线、角度、背景等可能会失效

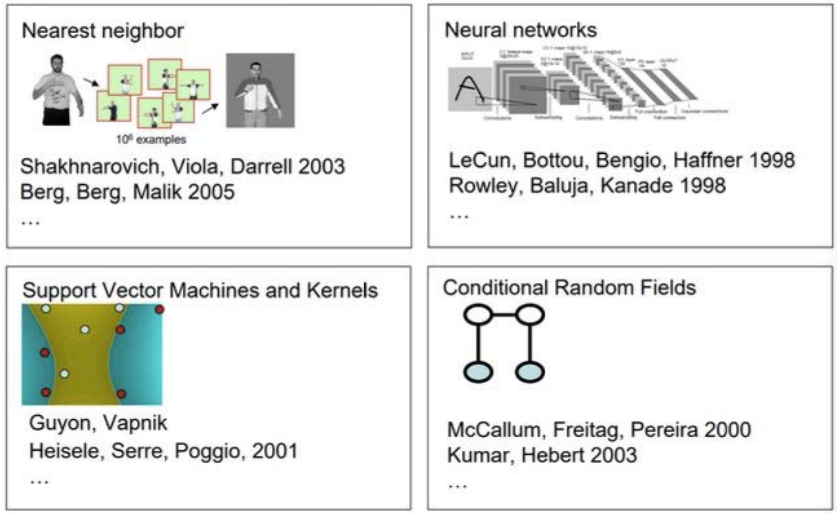
2）可扩展性差，需要为每一种物体设计专门的算法

数据驱动！

数据驱动的图像分类方法



1）收集大量图片



2）用机器学习训练分类器

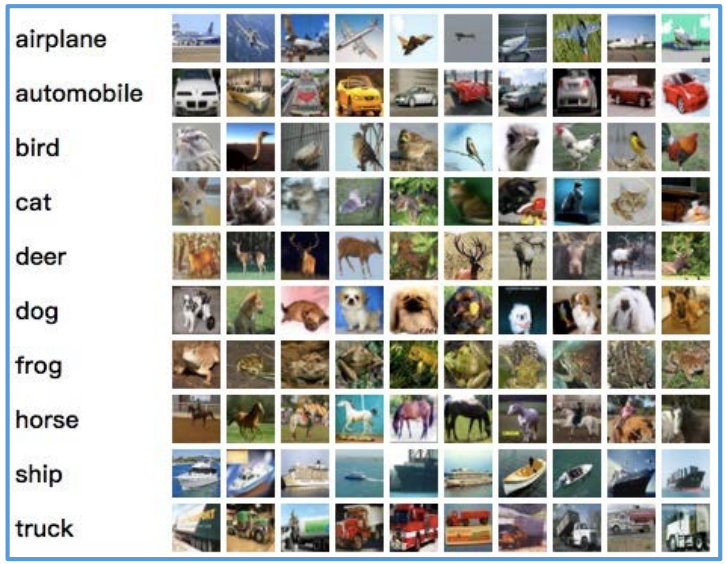
3）用新图片评价分类器

图像分类入门 / benchmark数据集

MNIST

训练集 60000
测试集 10000

CIFAR-10
（CIFAR-100）



32×32小图像，10类
训练集 50000
测试集 10000

ILSVRC

10,000,000+ 图片
10,000+ 类别

香草分类器：Nearest Neighbor

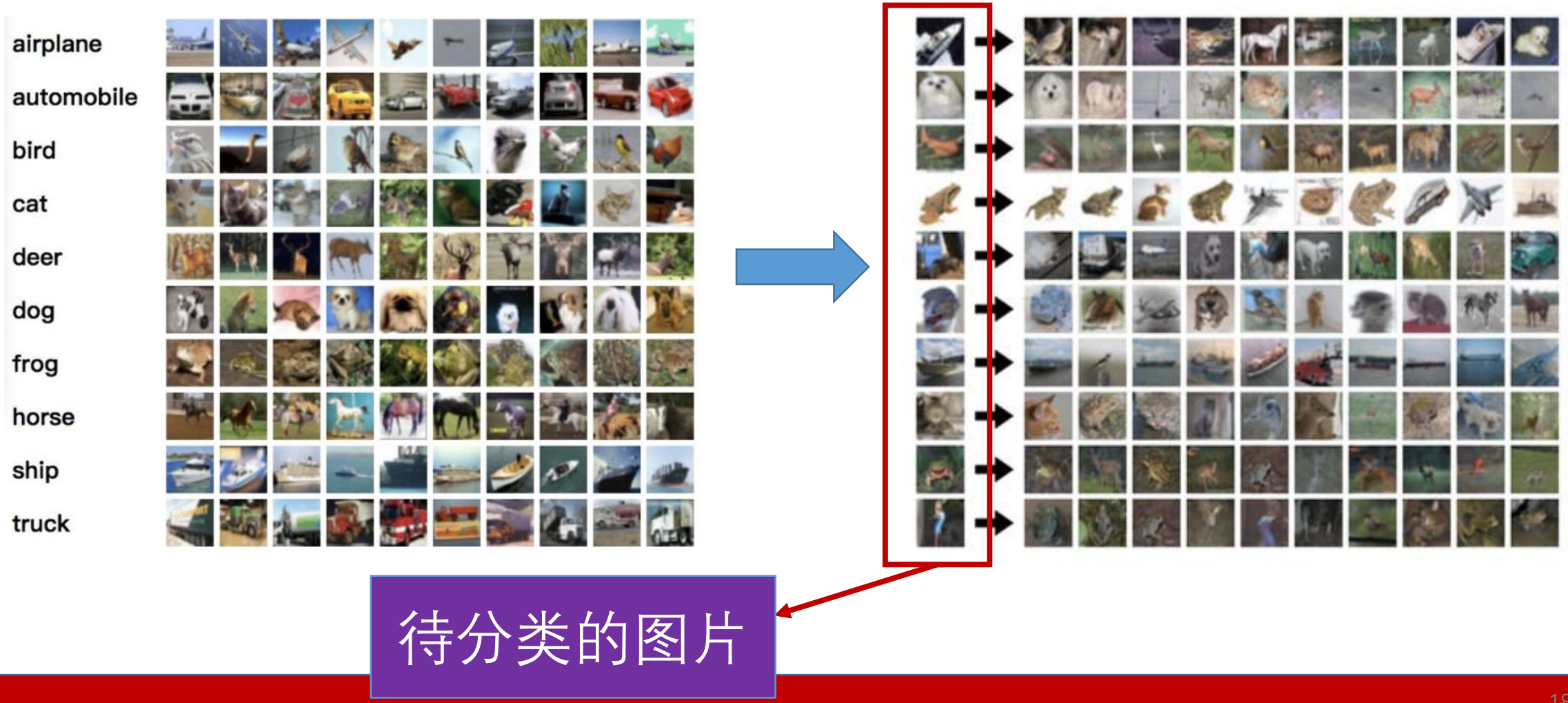
```
def train(images, labels):
    # 机器学习算法
    return NNmodel
```

记住（存储）所有图像及其标签

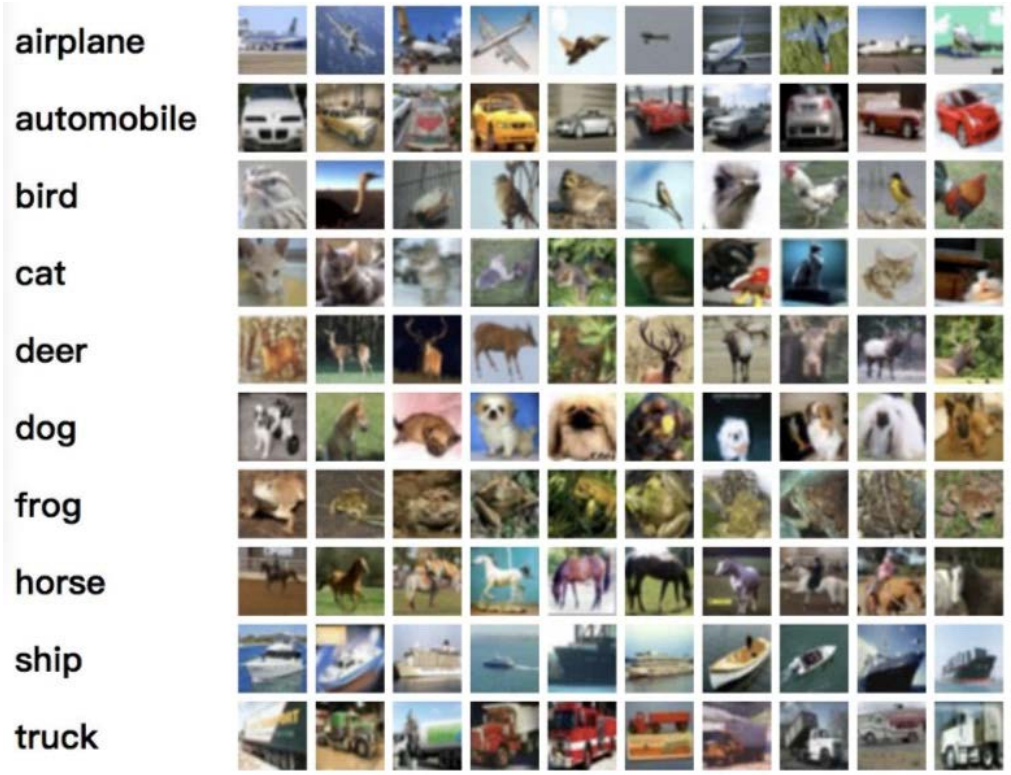
```
def predict(NNmodel, new_images):
    # 使用NNmodel预测new_images的标签
    return predicted_labels
```

使用距离最近的图像的标签来预测新图像的标签

NN示例：CIFAR10数据集



NN示例：CIFAR10数据集



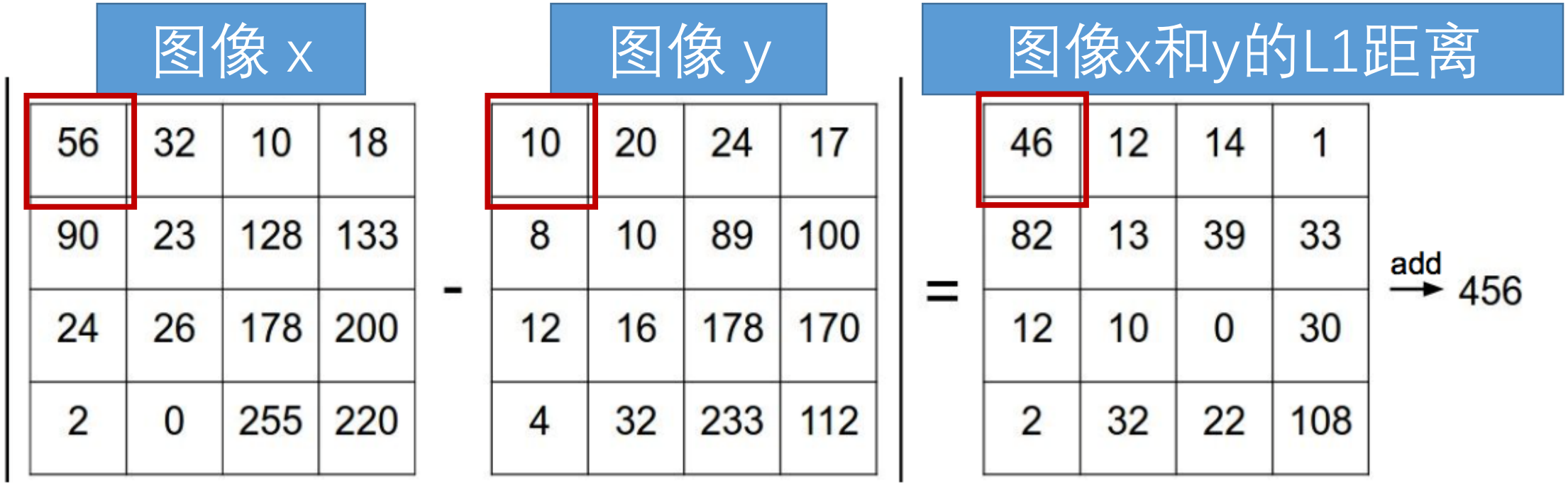
距离最近的10个图片

待分类的图片

图片距离方程

L1 distance或曼哈顿（Manhattan）距离

$$d(x,y) = \sum_{i=1}^{n} |x_i - y_i|$$



Nearest Neighbor分类器代码示例

```python
import numpy as np

class NearestNeighbor(object):
    """NN分类器"""
    def __init__(self):
        pass

    def train(self, X, y):
        """X为N个训练图片，每个图片D维；y是大小为N的一维向量"""
        self.X_train = X
        self.y_train = y

    def predict(self, X):
        """X为N个待预测图片"""
        numberOfPred = X.shape[0]
        y_pred = np.zero(numberOfPred, dtype = self.y_train.dtype)

        for i in range(numberOfPred):
            distances = np.sum(np.abs(self.X_train - X[i, :]), axis = 1)
            min_idx = np.argmin(distances)
            y_pred[i] = self.y_train[min_idx]

        return y_pred

```

记住（存储）所有图像及其标签

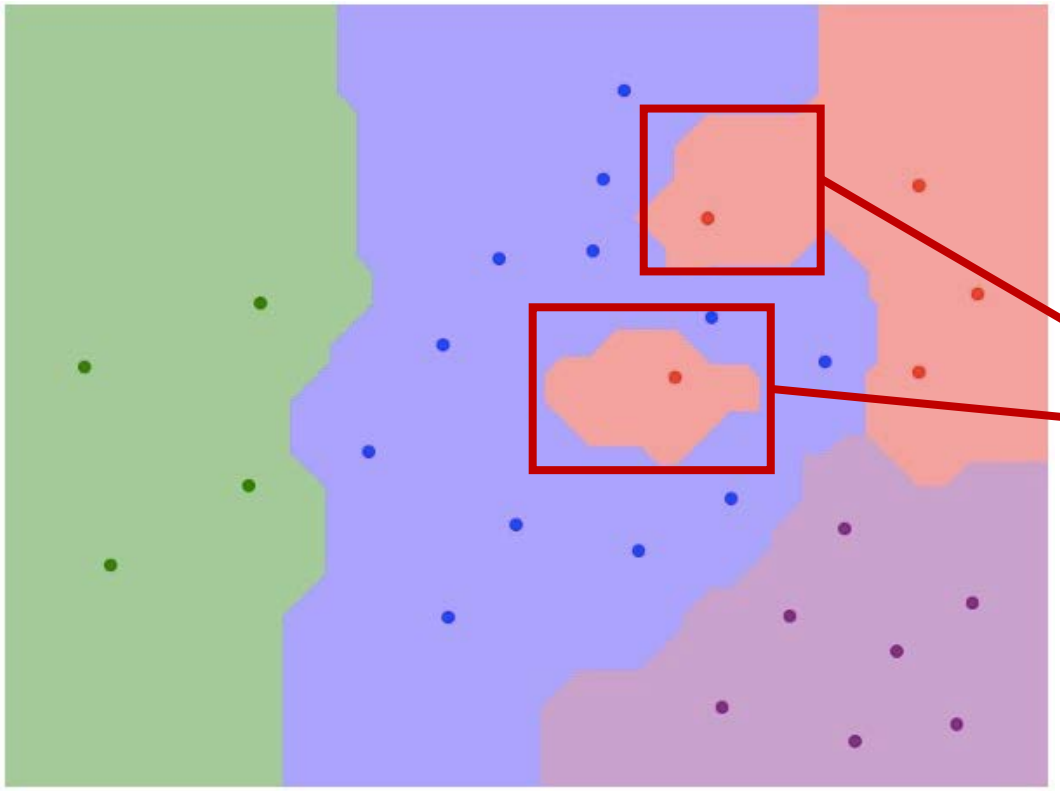
使用距离最近的图像的标签来预测新图像的标签

Nearest Neighbor的问题

```python
import numpy as np

class NearestNeighbor(object):
    """NN分类器"""
    def __init__(self):
        pass

    def train(self, X, y):
        """X为N个训练图片，每个图片D维；y是大小为N的一维向量"""
        self.X_train = X
        self.y_train = y

    def predict(self, X):
        """X为N个待预测图片"""
        numberOfPred = X.shape[0]
        y_pred = np.zero(numberOfPred, dtype = self.y_train.dtype)

        for i in range(numberOfPred):
            distances = np.sum(np.abs(self.X_train - X[i, :]), axis = 1)
            min_idx = np.argmin(distances)
            y_pred[i] = self.y_train[min_idx]

        return y_pred

```

时间复杂度：
训练O(1)，测试O(N)

NN算法的效果演示



Metric

L1 L2

Num Neighbors (K)

1 2 3 4 5 6 7

Num classes

2 3 4 5

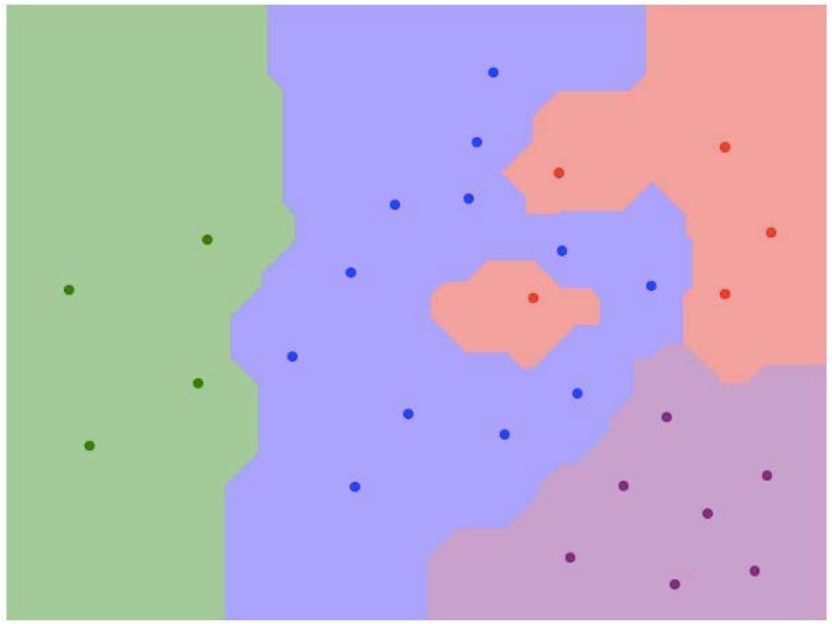
Num points

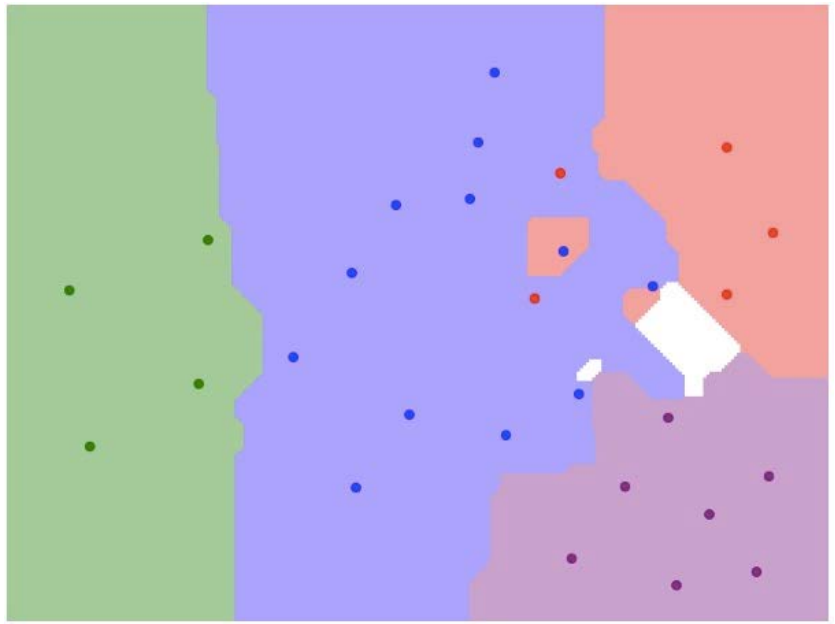
20 30 40 50 60

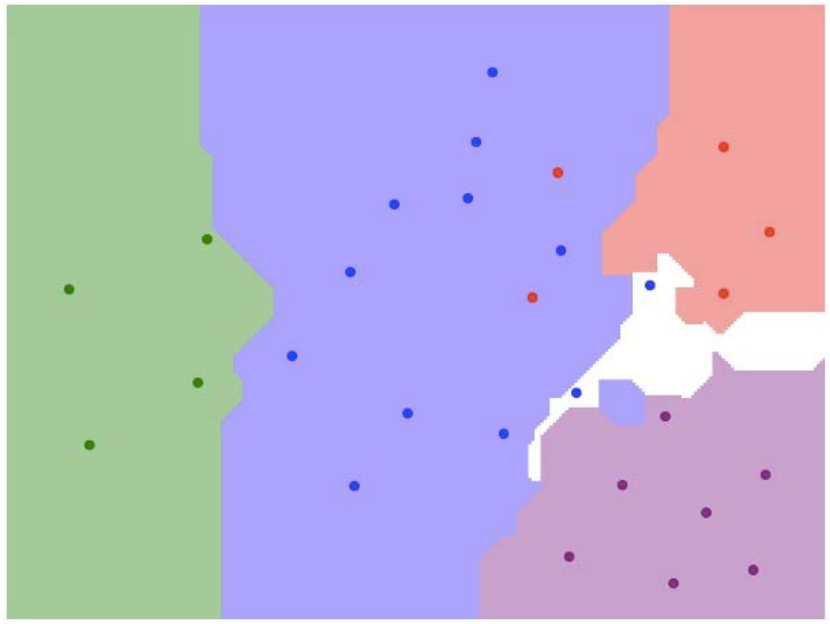
Outlier导致分类错误

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

从NN到KNN（K-Nearest Neighbors）



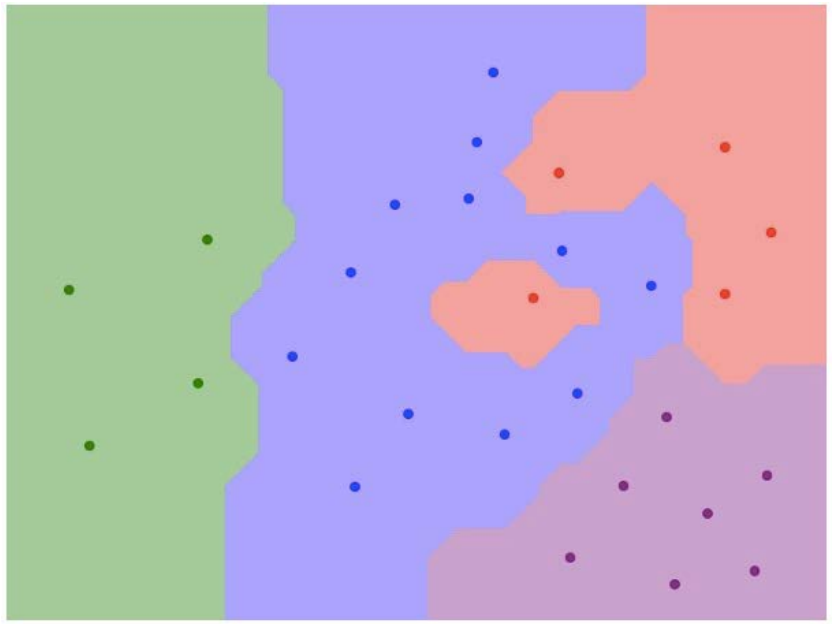


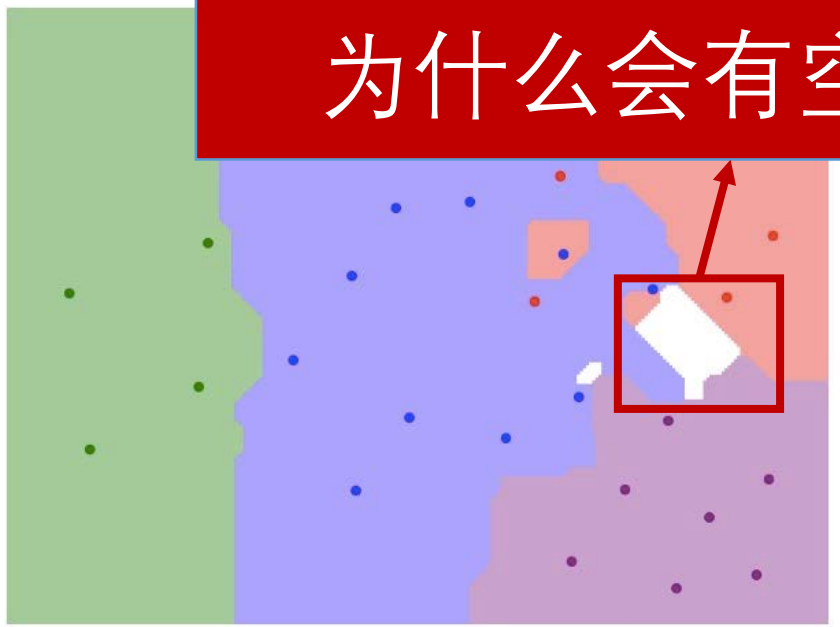


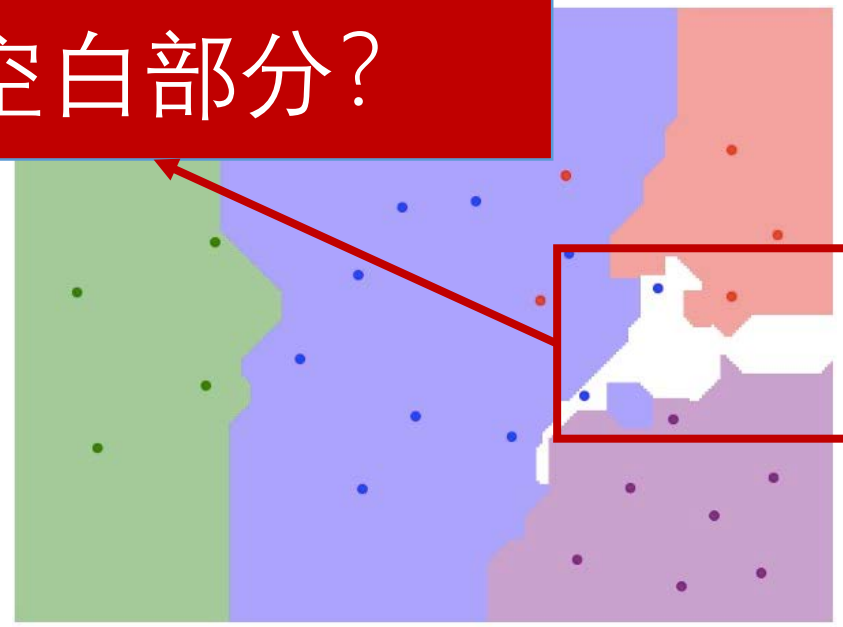
从K个距离最近的邻居中找占据多数的类别作为预测结果

从NN到KNN（K-Nearest Neighbors）

为什么会有空白部分？







从K个距离最近的邻居中找占据多数的类别作为预测结果

图片距离方程的影响

L1 distance

曼哈顿（Manhattan）距离

$$d(x,y)=\sum_{i=1}^{n}|x_i-y_i|$$

L2 distance

欧氏（Euclidean）距离

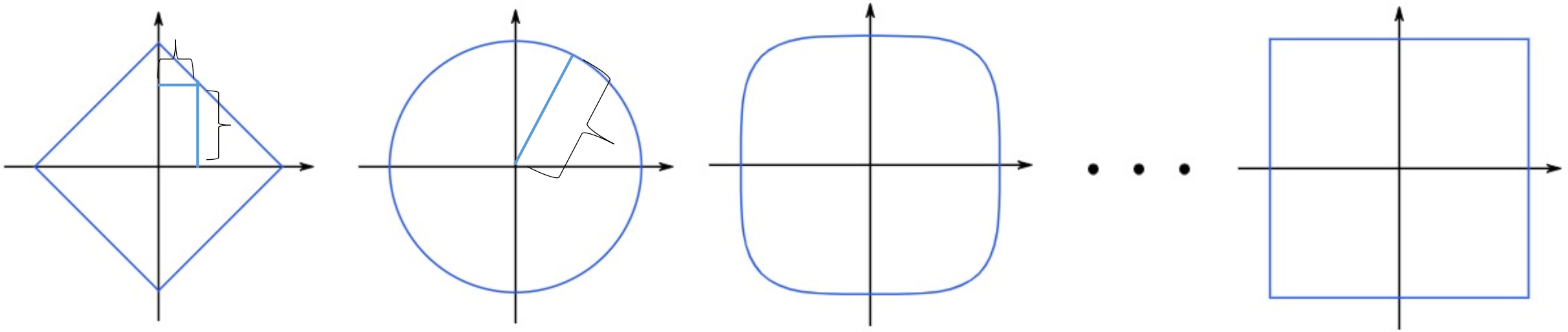
$$d(x,y)=\sqrt{\sum_{i=1}^{n}(x_i-y_i)^2}$$

明式（Minkowski）距离

$$d(x,y)=\left(\sum_{i=1}^{n}|x_i-y_i|^p\right)^{1/p}$$

图片距离方程的影响：明式距离

$$d(x,y)=\sum_{i=1}^{n}|x_i-y_i| \qquad d(x,y)=\sqrt{\sum_{i=1}^{n}(x_i-y_i)^2}$$

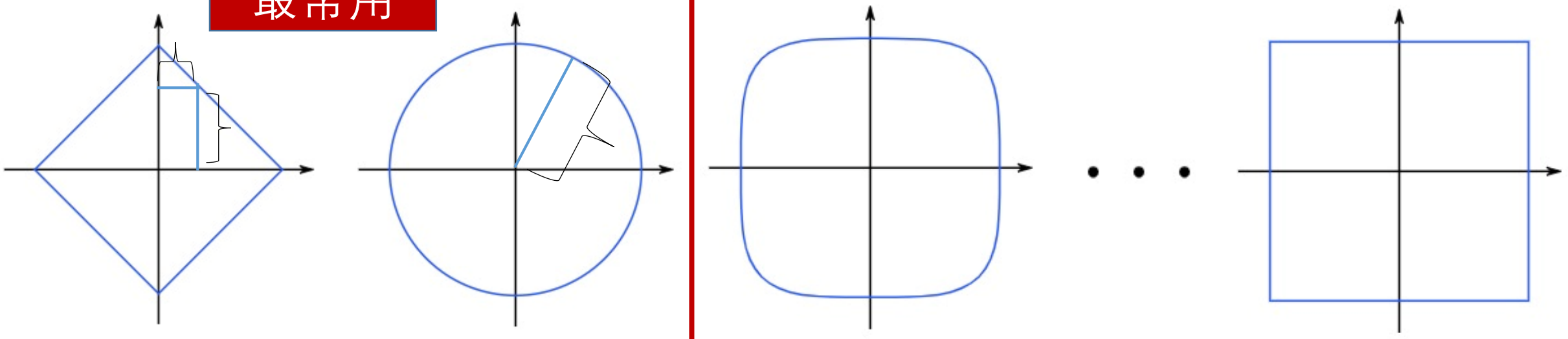


L1 distance　　L2 distance　　L3 distance　　L∞ distance

图片距离方程的影响：明式距离

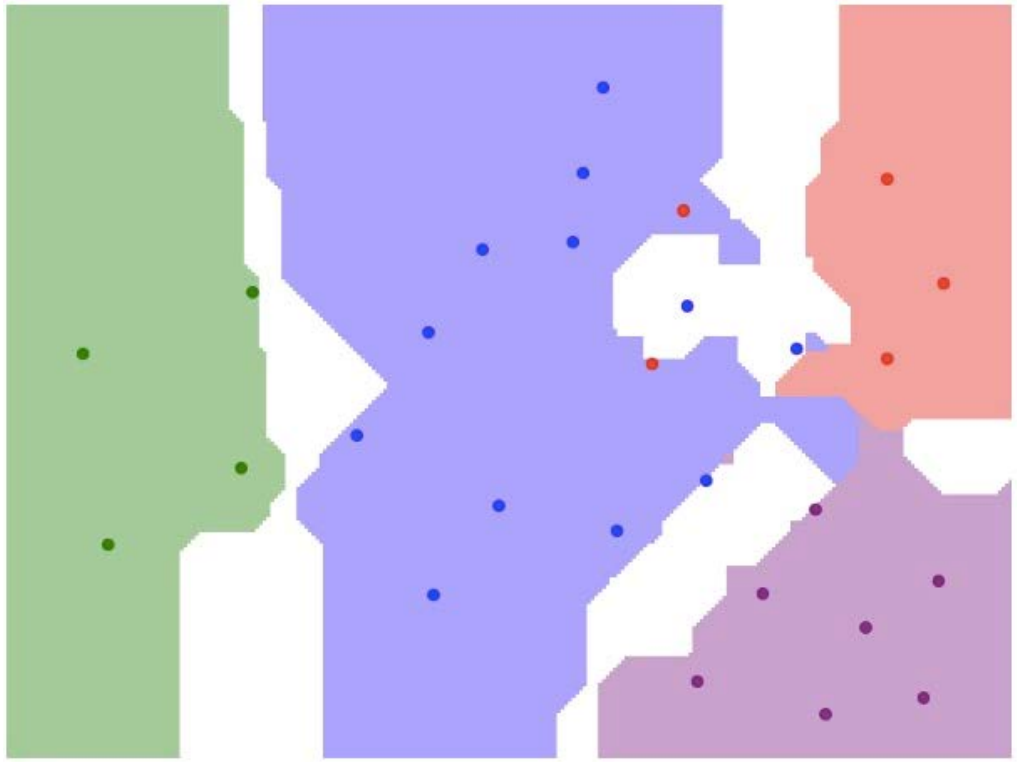
$d(x,y)=\sum_{i=1}^{n}|x_i-y_i|$　　$d(x,y)=\sqrt{\sum_{i=1}^{n}(x_i-y_i)^2}$

最常用

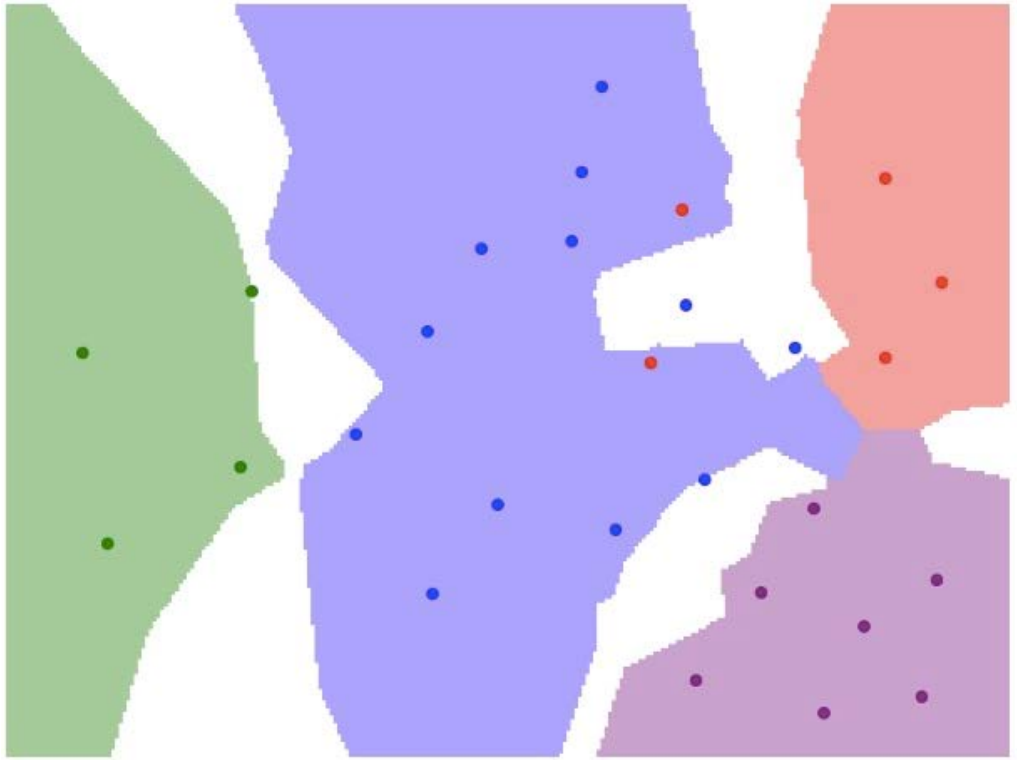


L1 distance　　L2 distance　　L3 distance　　L∞ distance

图片距离方程的影响： L1 Vs. L2



Metric
L1 L2
Num Neighbors (K)
1 2 3 4 5 6 7
Num classes
2 3 4 5
Num points
20 30 40 50 60



Metric
L1 L2
Num Neighbors (K)
1 2 3 4 5 6 7
Num classes
2 3 4 5
Num points
20 30 40 50 60

超参数（Hyperparameters）

- 选取的邻居个数：K
- 选取的距离计算方式：d(x, y)

- 算法或者模型中需要人为设定的参数
  - ✓无法从数据中学习到最优值
  - ✓依赖于具体的问题，需要尝试各种值及其组合

- 如何设置最优的超参数?

寻找最优超参数

- 选项1：使模型在整个数据集上表现最好

超参数在现有数据集上完美分类，造成过拟合!

Entire dataset as training data

寻找最优超参数

- 选项1：使模型在整个数据集上表现最好

超参数在现有数据集上完美分类，造成过拟合!

- 选项2：使模型在测试集上表现最好

无法知道超参数在新数据集上的表现

寻找最优超参数

- 选项1：使模型在整个数据集上表现最好

超参数在现有数据集上完美分类，造成过拟合！

- 选项2：使模型在测试集上表现最好

无法知道超参数在新数据集上的表现

- 选项3：使模型在验证集上表现最好，同时报告测试集表现

寻找最优超参数

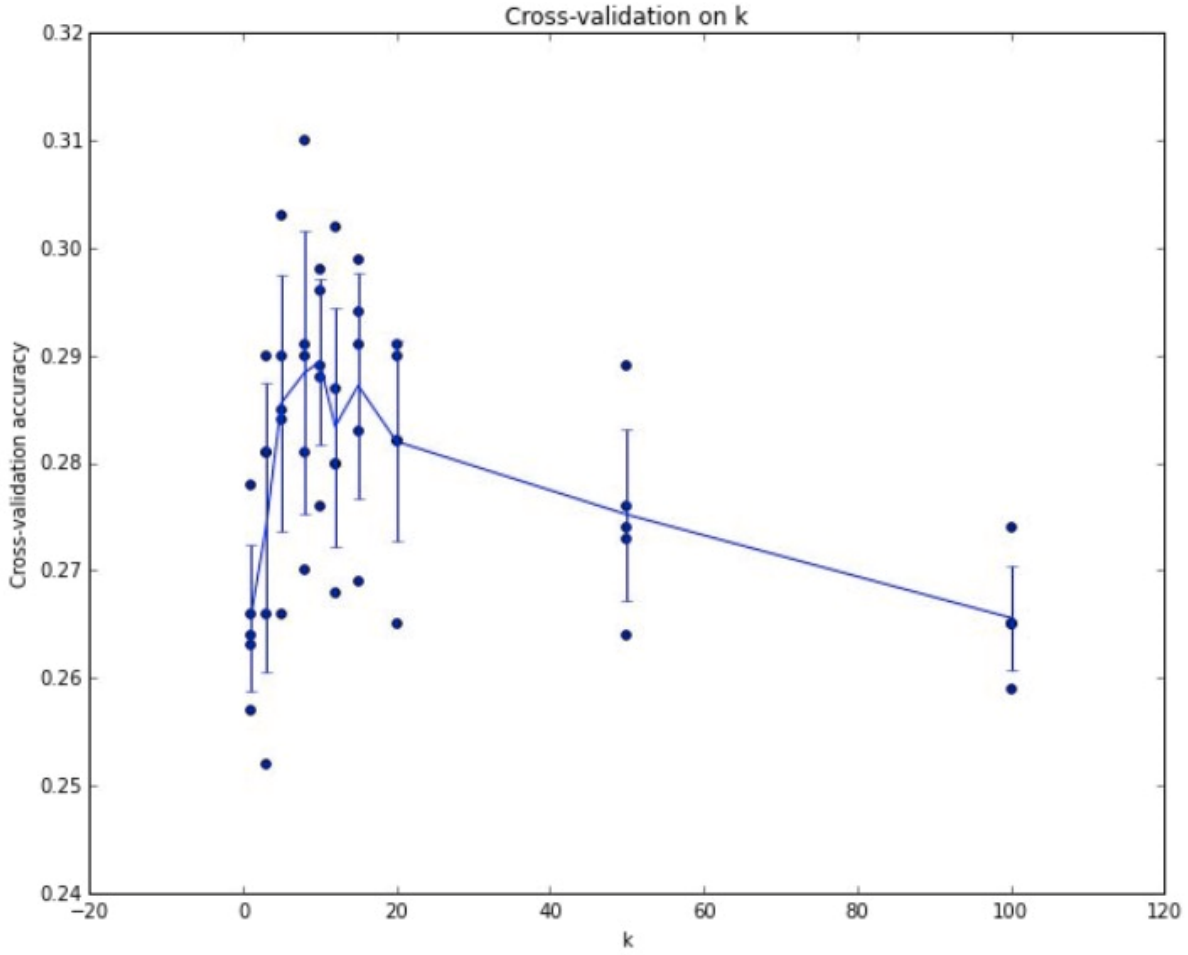
5折交叉验证

- 选项4：交叉验证（cross validation）
  - ✓对于大型数据集和深度学习较少使用

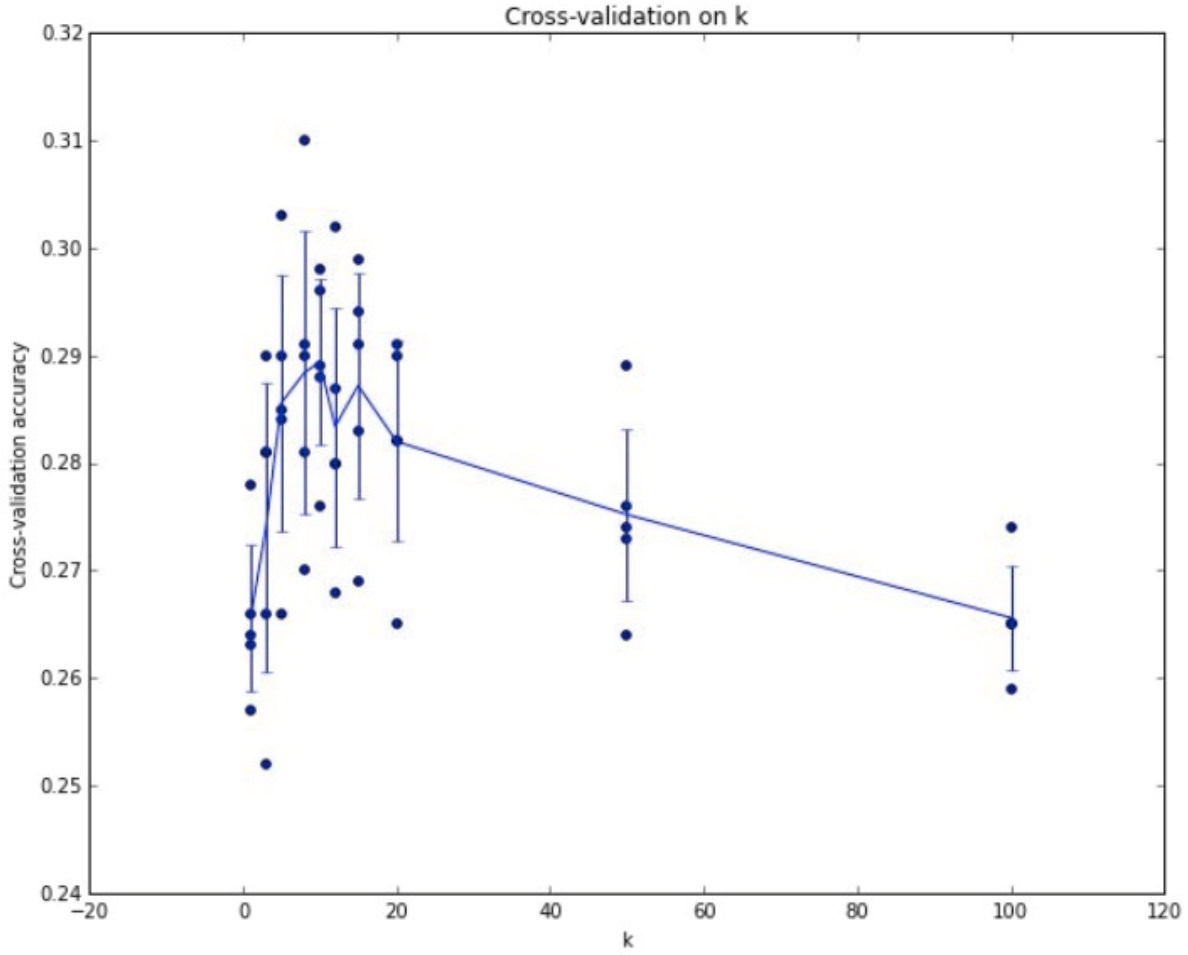
Fold1	Fold2	Fold3	Fold4	Validation	Test
Fold1	Fold2	Fold3	Validation	Fold5	Test
Fold1	Fold2	Validation	Fold4	Fold5	Test
Fold1	Validation	Fold3	Fold4	Fold5	Test
Validation	Fold2	Fold3	Fold4	Fold5	Test

超参数选择的可视化

1）使用cross-validation



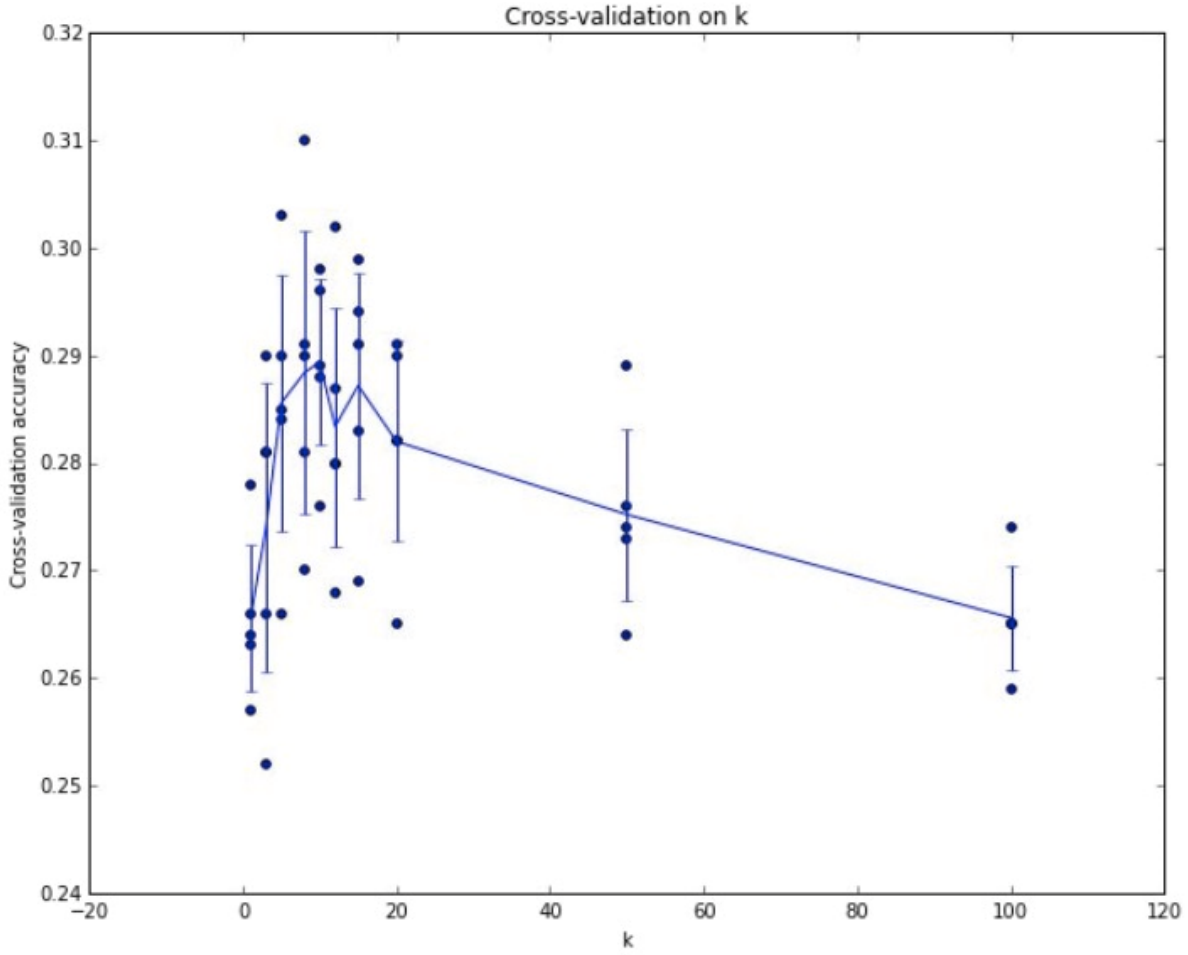
超参数选择的可视化



1）使用cross-validation

2）不断增大K，对于每个K，计算验证集上平均accuracy和standard deviation

超参数选择的可视化



1）使用cross-validation

2）不断增大K，对于每个K，计算验证集上平均accuracy和standard deviation

3）一般选取accuracy值较大，且表现较稳定的K

(K)NN算法小结

- 优点：简单

- 缺点
  - ✓训练阶段是简单的标签记忆(non-parametric)
  - ✓像素距离和图像信息的语义鸿沟
  - ✓对训练集数据分布要求较高

1）预测效率低下，O(N)时间复杂度
2）lazy learner，训练阶段不做任何泛化；而图片分类需要具备泛化能力的分类器（eager learner）

图片像素距离相近≠图像信息相近

训练集需要在整个像素空间中均匀分布，导致curse of dimensionality

实际应用中不会使用KNN做图像分类，但我们在第一次作业中实现这个算法，体会一下效果。

线性分类器： Linear Classifier

f(x, W)

狗的分数

猫的分数

狼的分数

64×64×3=12288个RGB数值

取分数最高的类别为预测类别

线性分类器： Linear Classifier

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

64×64×3=12288个RGB数值

$f(x, W)$

线性函数
x为图像RGB数值向量
W为权重矩阵（Weights）
Parametric approach

狗的分数

猫的分数

狼的分数

取分数最高的类别为预测类别

线性分类器： Linear Classifier

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

12288×1　偏置项，3×1
可理解为w_0

3×1　　3×12288

64×64×3=12288个RGB数值

$f(x, W)$

狗的分数

猫的分数

狼的分数

线性函数
x为图像RGB数值向量
W为权重矩阵（Weights）
Parametric approach

取分数最高的类别为预测类别

Linear Classification 算法示例

拉伸成一维向量

1.5	1.3	2.1	0.0
0.2	-0.5	0.1	2.0
0.0	0.25	0.2	-0.3

W

56
231
24
2

x

\+

3.2
1.1
-1.2

b

=

437.9	狗分数
-96.8	猫分数
61.95	狼分数

Linear Classification 算法示例

拉伸成一维向量

狗的权重模板

图像表示

1.5	1.3	2.1	0.0
0.2	-0.5	0.1	2.0
0.0	0.25	0.2	-0.3

W

56
231
24
2

x

+

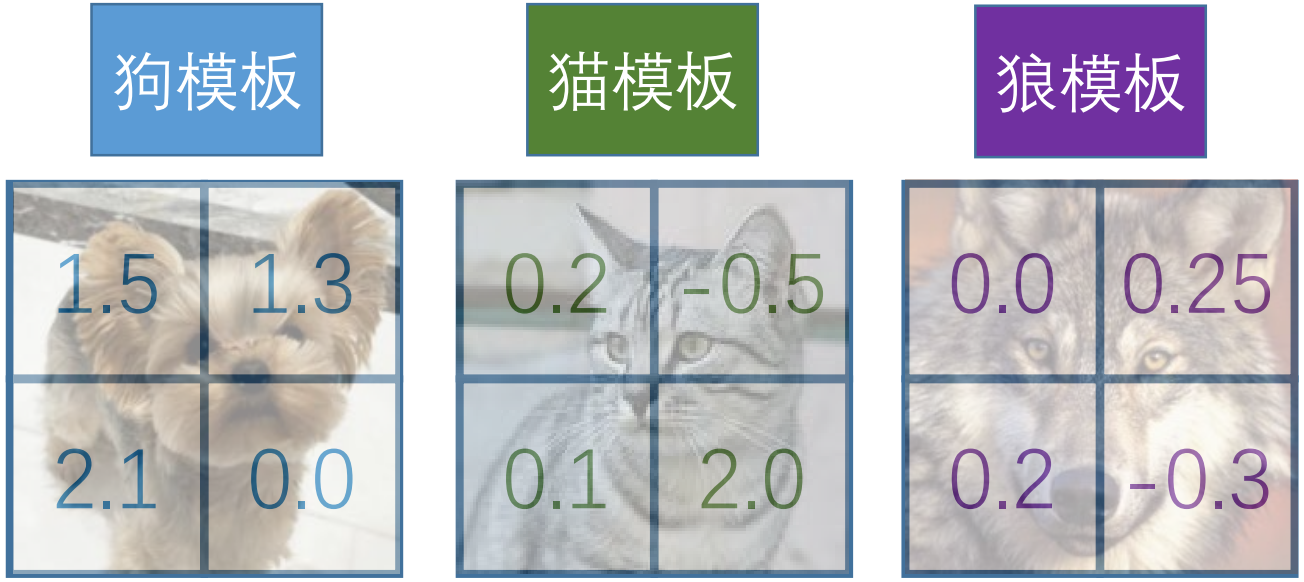
3.2
1.1
-1.2

b

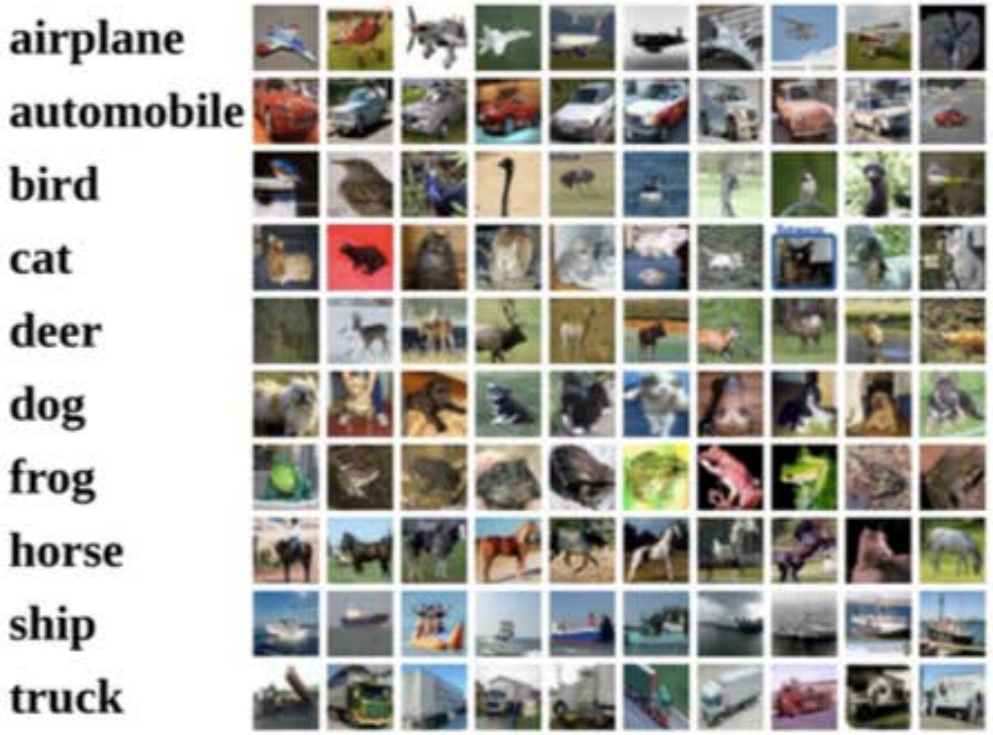
=

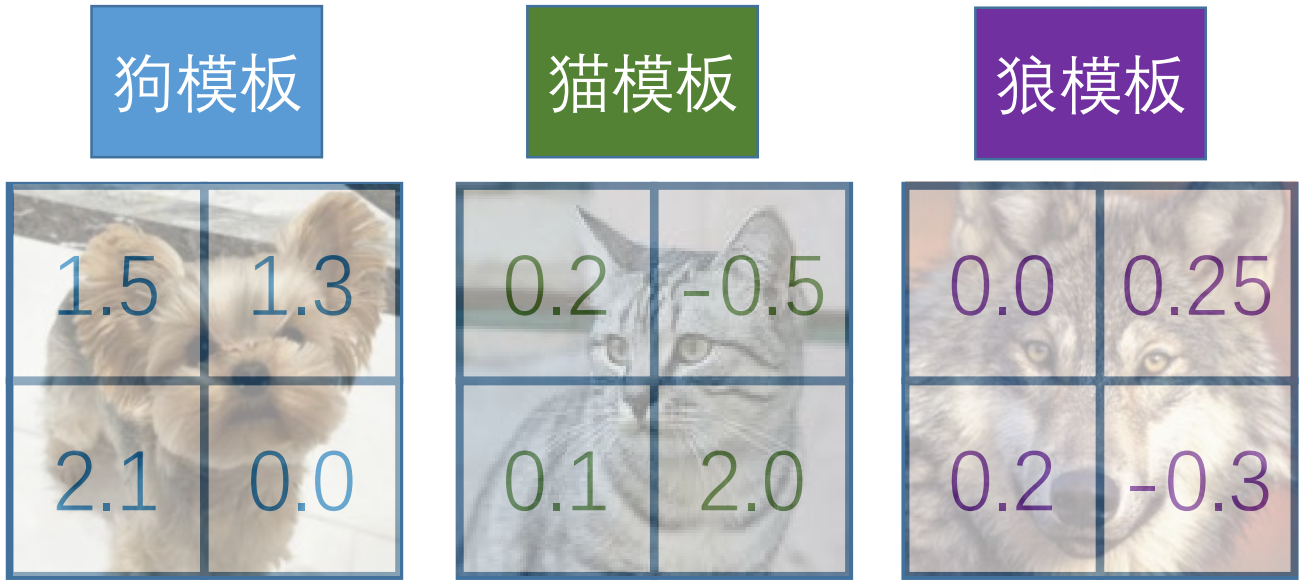
437.9	狗分数
-96.8	猫分数
61.95	狼分数

解释线性分类器：感官视角



解释线性分类器：感官视角

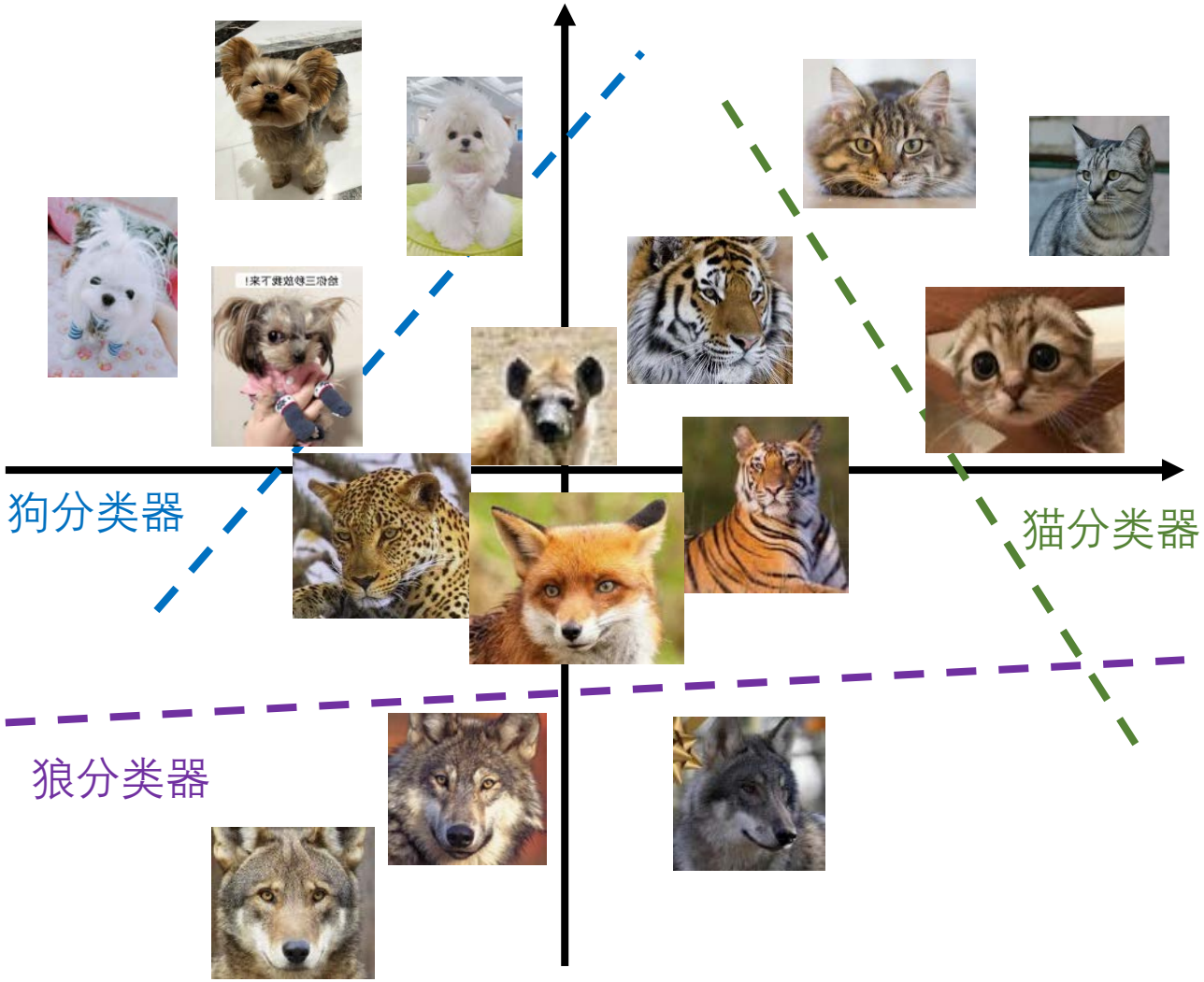




解释线性分类器：几何视角

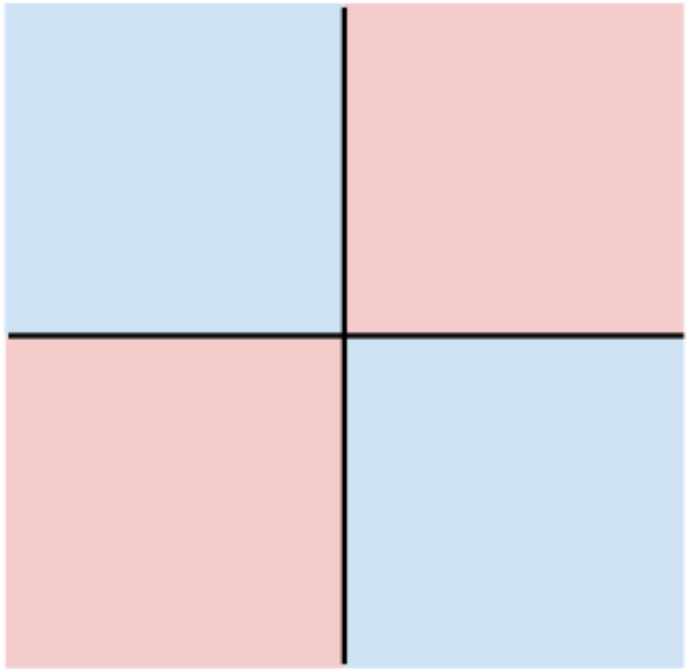
权重矩阵每一行
代表一个超平面

1.5	1.3	2.1	0.0
0.2	-0.5	0.1	2.0
0.0	0.25	0.2	-0.3

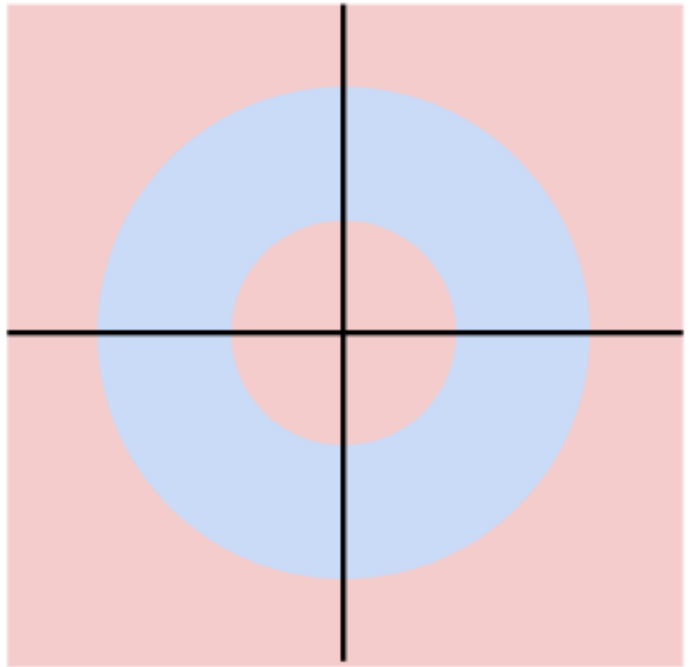


线性分类器很难适用的数据分布

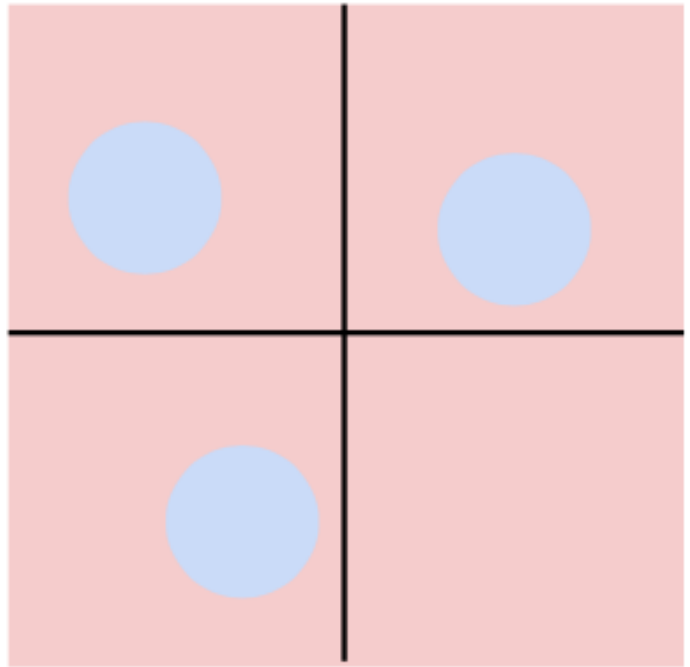
离散象限



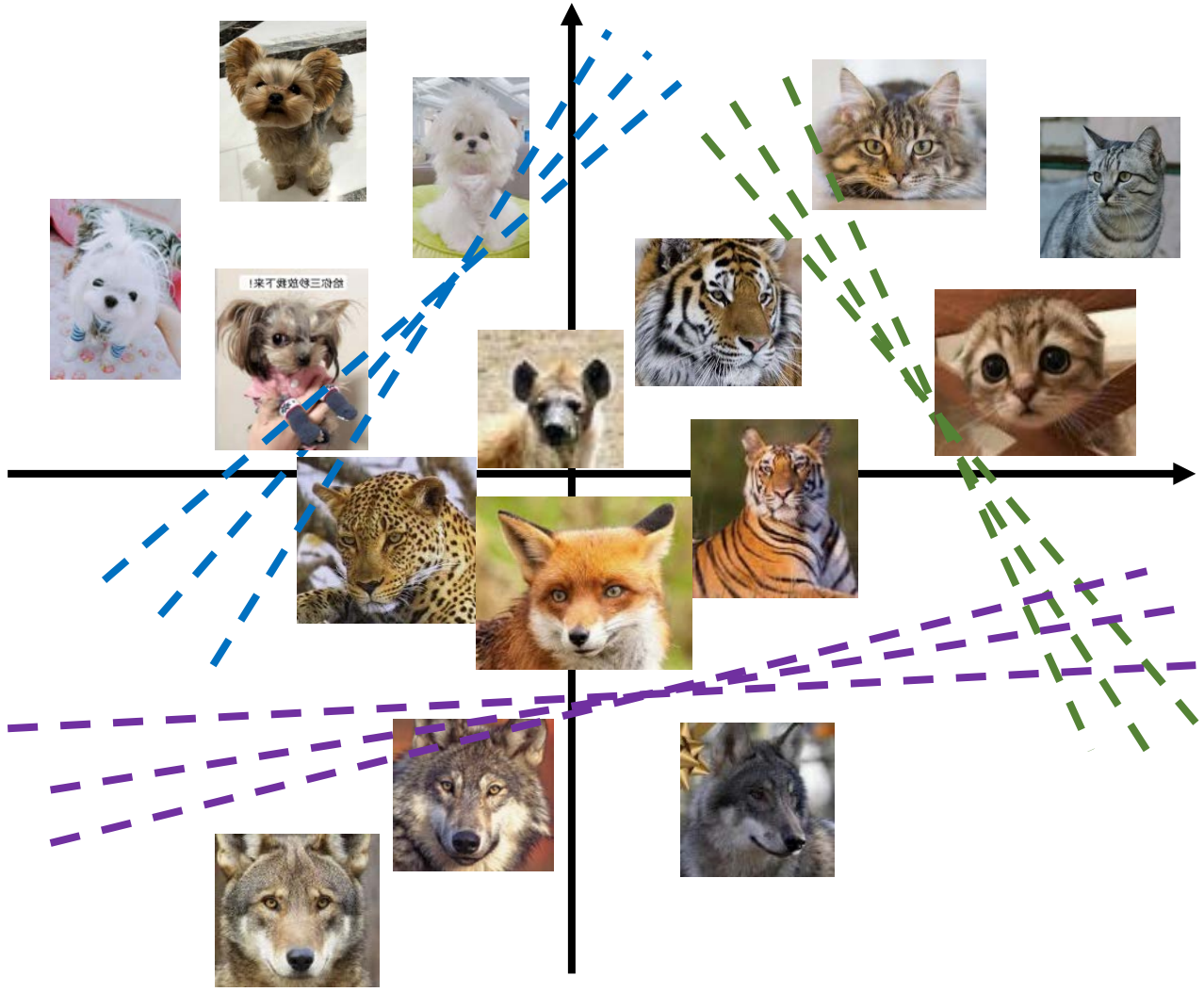
1<L2距离<2



孤岛数据



寻找最优权重矩阵



1.5	1.3	2.1	0.0
0.2	-0.5	0.1	2.0
0.0	0.25	0.2	-0.3

如何评价一个权重矩阵的好坏?

损失函数和参数优化（下次课）

Linear Classifier 代码示例

```python
class LinearClassifier(object):

    def __init__(self):
        self.W = None

    def train(self, X, y, learning_rate=1e-3, reg=1e-5, num_iters=100,
              batch_size=200):
        """
        Inputs:
        - X: N*D的训练数据
        - y: N*1的标签向量
        Outputs:
        打印每次迭代的loss
        """
        num_train, dim = X.shape
        num_classes = np.max(y) + 1
        if self.W is None:
            self.W = 0.001 * np.random.randn(dim, num_classes)

        loss_history = []
        for it in range(num_iters):
            # *****训练代码*****

        return loss_history

    def predict(self, X):
        """
        Inputs:
        - X: N*D的待预测数据
        Outputs:
        - y_pred: N*1的预测标签向量
        """
        y_pred = np.zeros(X.shape[0])
        # *****预测代码*****

        return y_pred
```

Weight矩阵初始化

计算损失函数
优化Weight矩阵

预测测试集标签

小结

- 图像分类任务
  - ✓CV的核心任务
  - ✓面临的挑战
- 传统方法
- 数据驱动的方法
  - ✓KNN
  - ✓Linear classifier

L03：损失函数和优化