K-Nearest Neighbors (KNN)

The “hello world” Algorithm

2019年6月26日

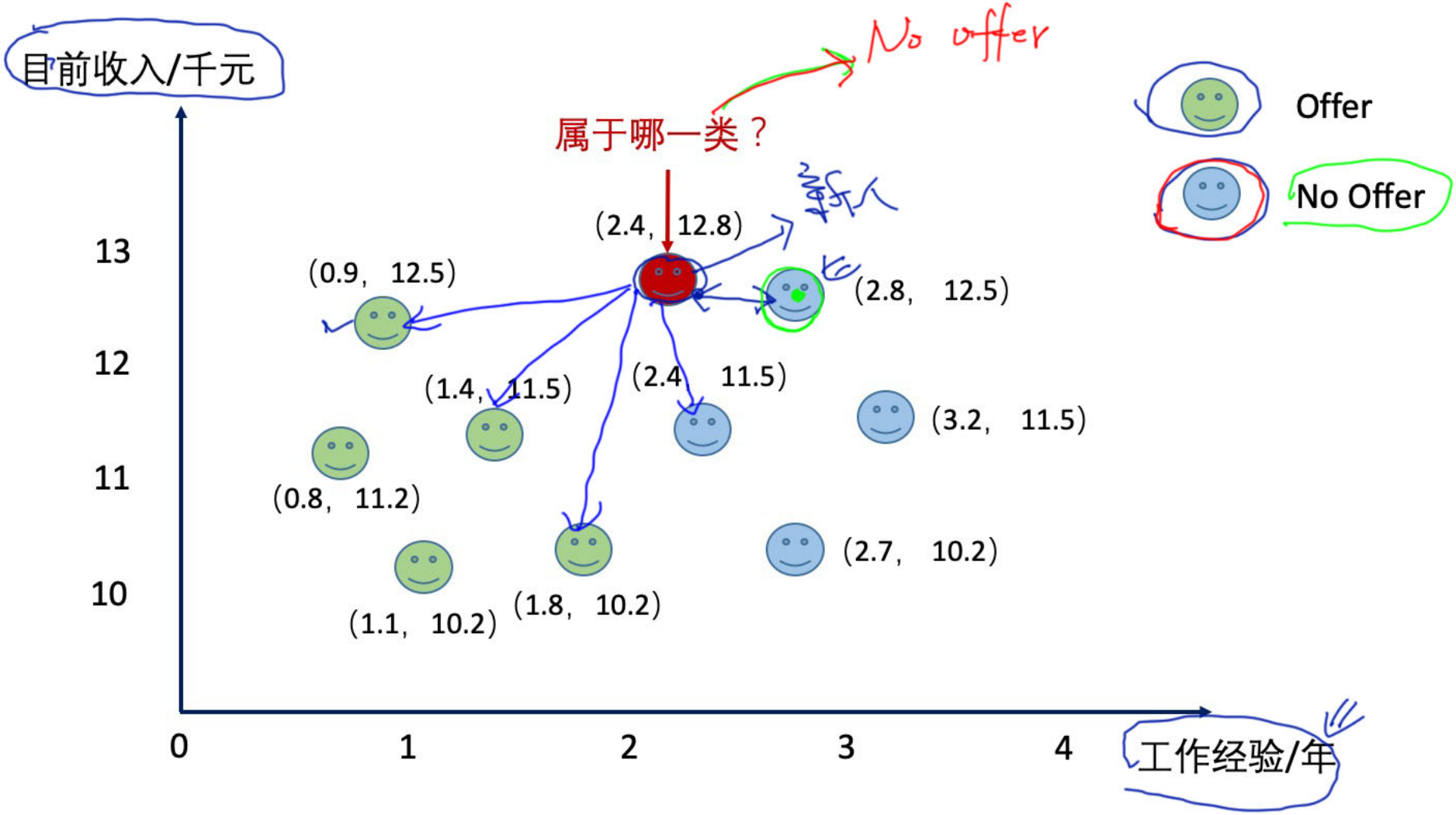
李文哲

K最近邻（KNN）算法

- 最容易**理解**的算法：**理解核心不超过5分钟**
- 最容易**实现**的算法：**从零实现不超过5行代码**

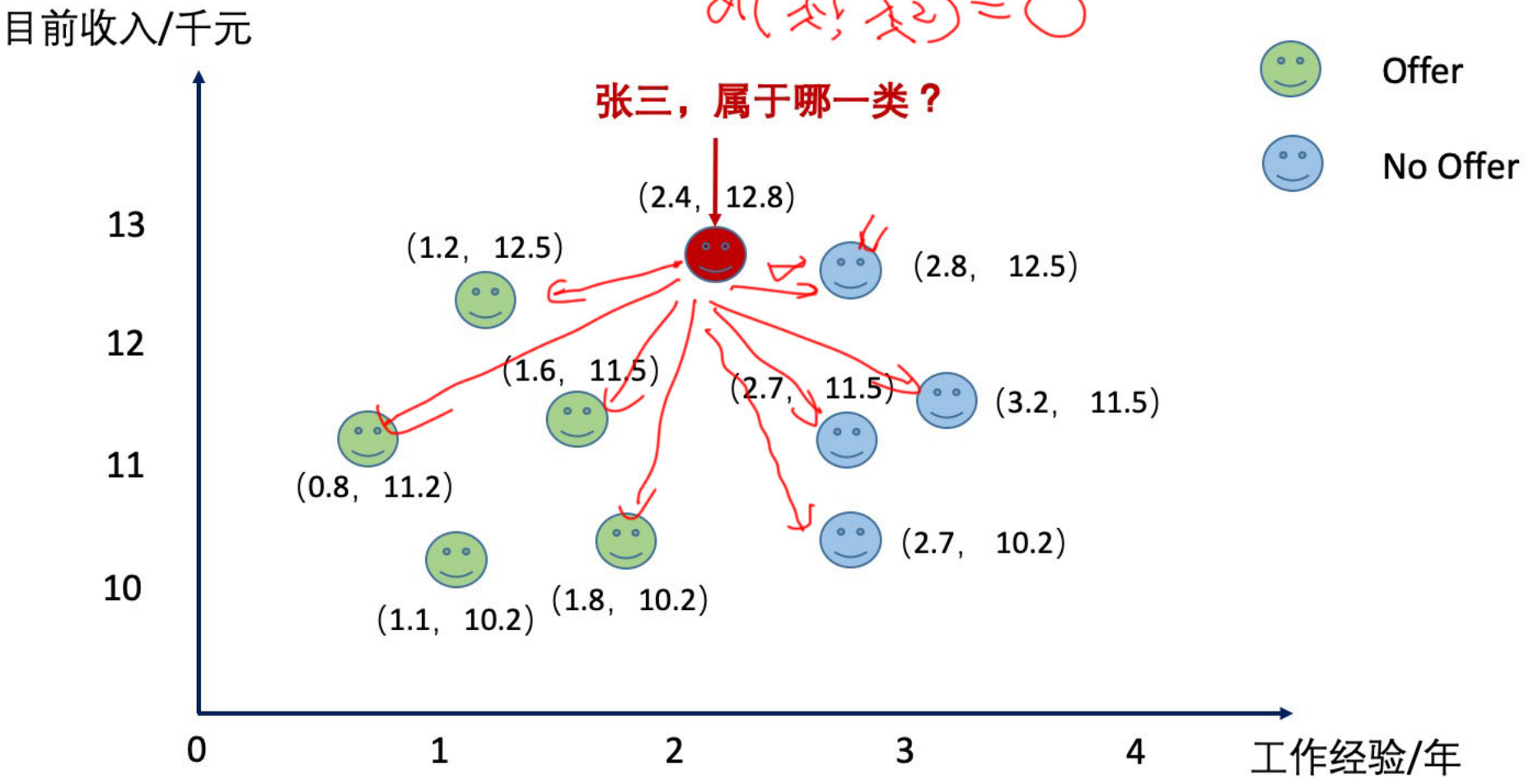
K最近邻（KNN）算法

Q：张三要参加一家公司的面试，但它通过各种渠道得知了拿到offer的人的情况和没有拿到offer人的情况。我们一起来预测一下张三是否会拿到offer?



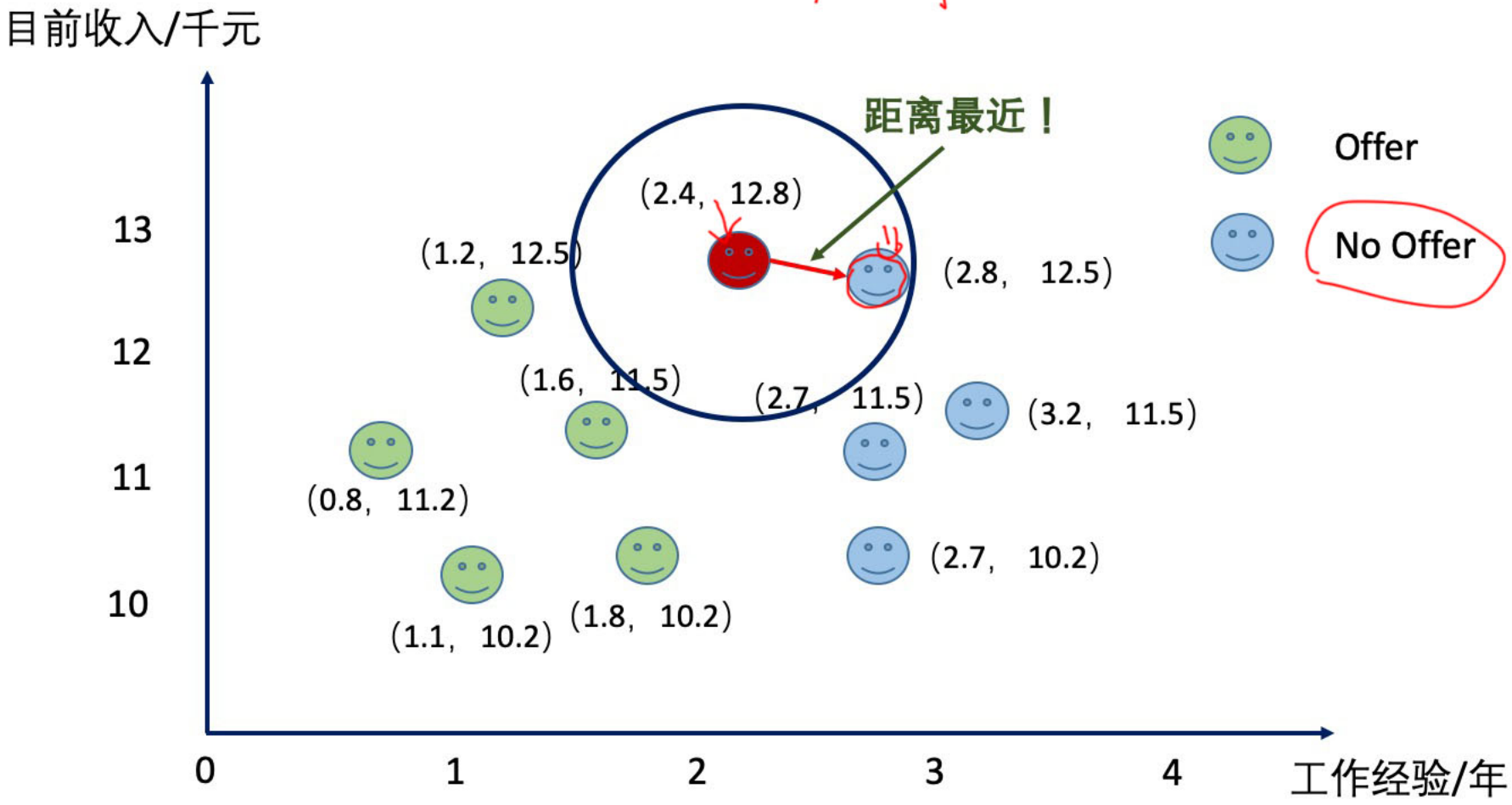
K最近邻（KNN）算法

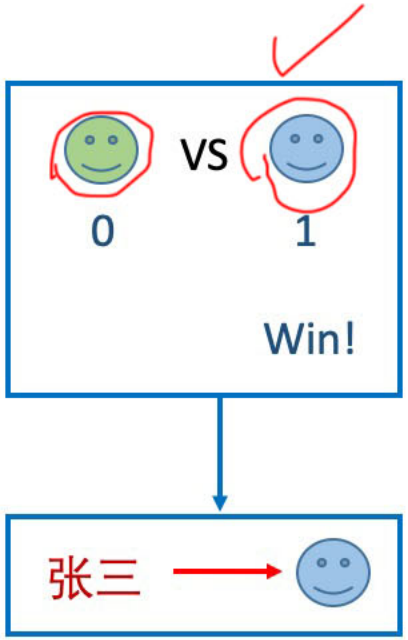
$d(x_1, x_2) = 0$



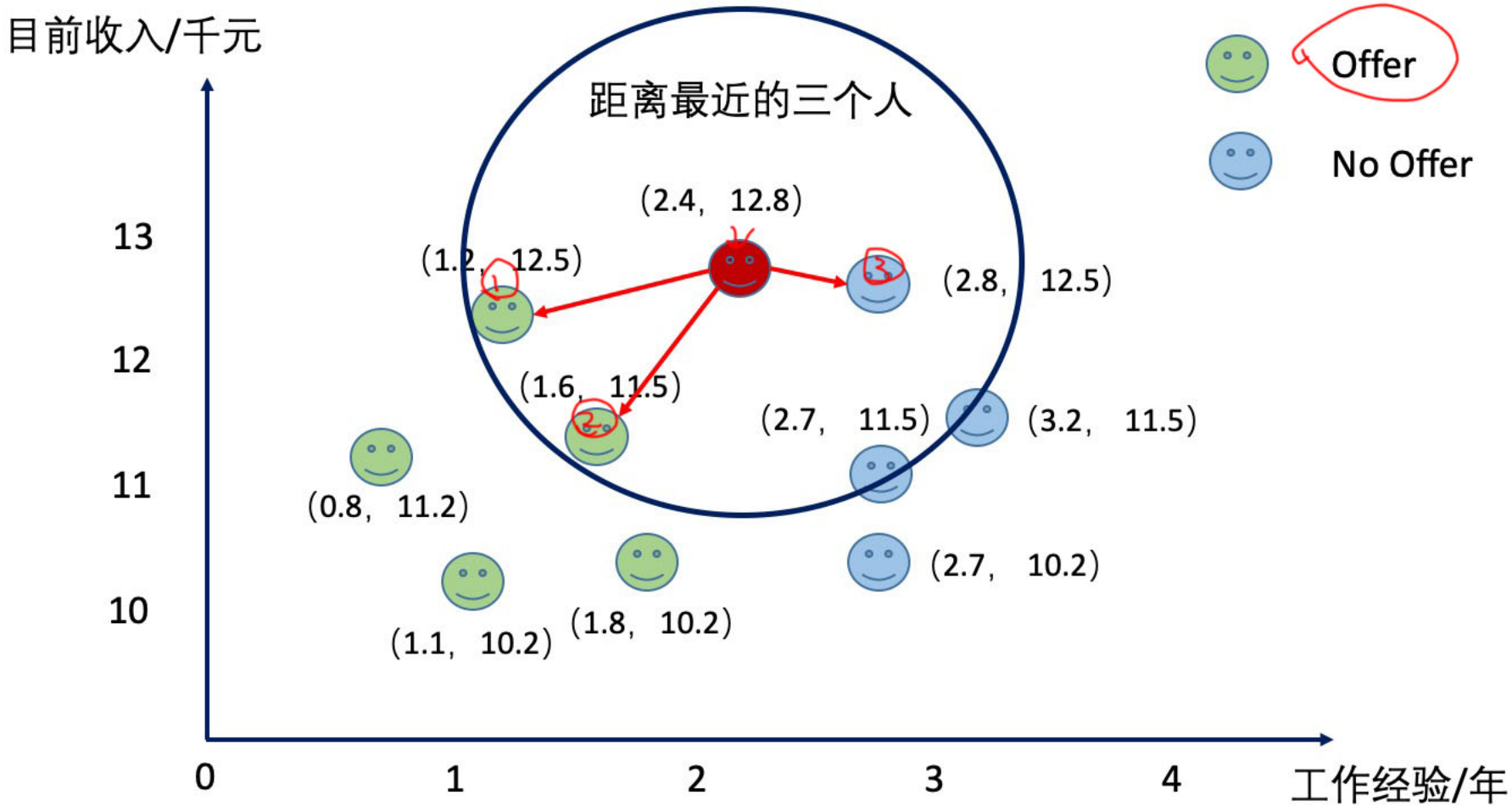
K最近邻（KNN）算法（K=1）

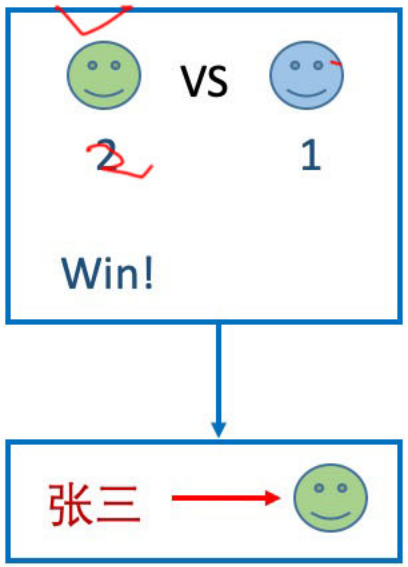
Top 1 points





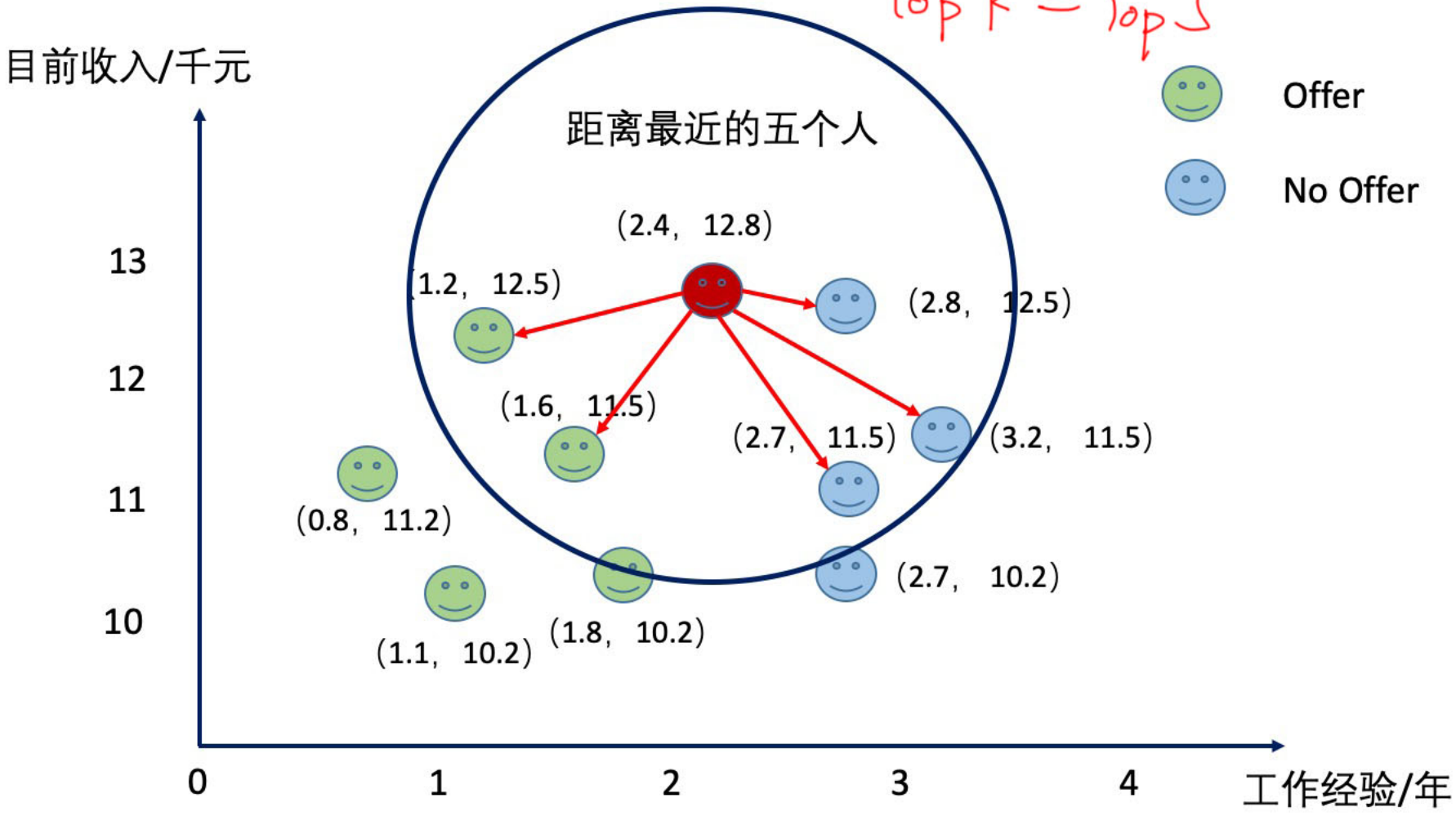
K最近邻（KNN）算法（K=3）



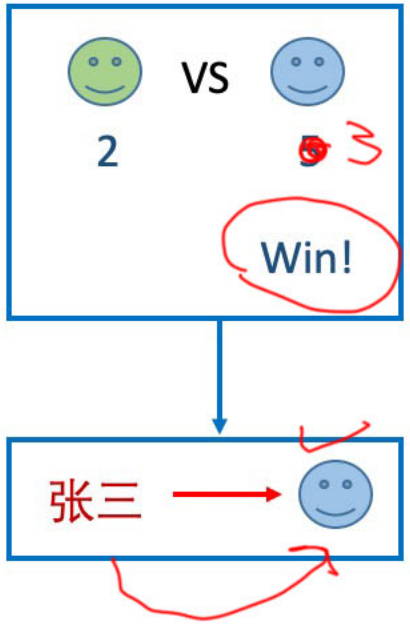


K最近邻（KNN）算法 (K=5)

Top K = Top 5



投票

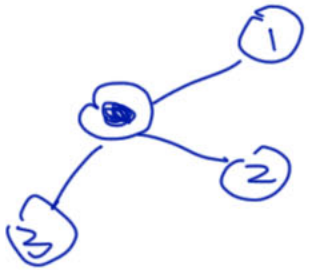


K最近邻（KNN）算法

问题：在使用KNN算法的时候，我们一般会选择**奇数（odd number）**的K, 为什么？

k=1, 3, 5, 7, 9 ...

k=3



⇒ 平分

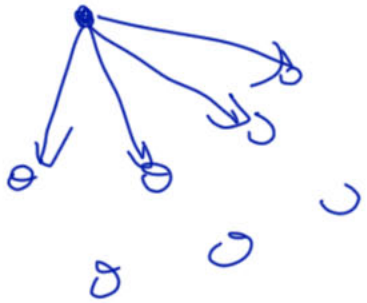
KNN:

clf.fit(x, y) x

内存分配

数据存下来

No Traing.



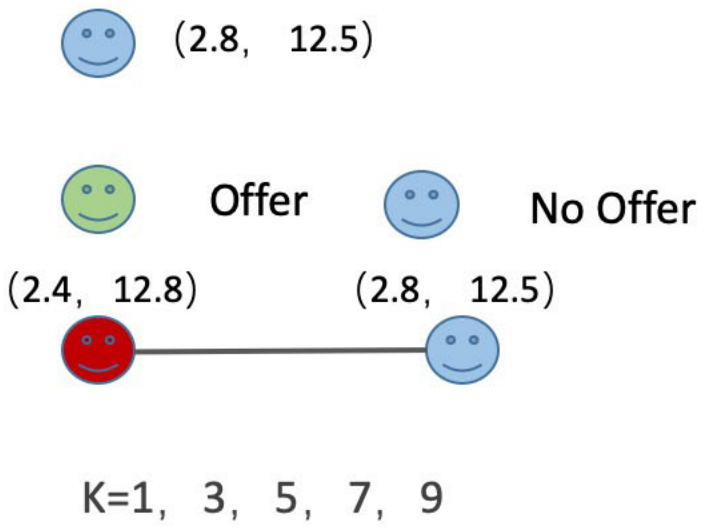
Coding Time–1

有几个需要考虑的问题

模型

kNN

1. 把一个物体表示成向量 / 矩阵
2. 标记号每个物体的标签（i.e., offer/no offer）
3. 计算两个物体之间的距离/相似度
4. 选择合适的K



K=1， 3， 5， 7， 9

1. 把一个物体表示成向量

- 这也叫做 **“特征工程”** 英文叫**Feature Engineering**
- 模型的**输入一定是数量化的信息**，我们需要把现实生活中的物体表示成**向量/矩阵/张量**形式。

特征

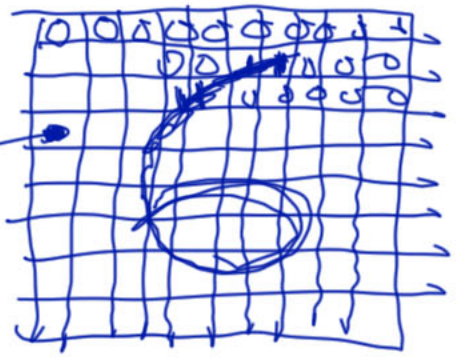
是否本科：1

人 = (年：25、性：男，收入：20k，身高：175cm，是否性：0，)

图片 6 = (矩阵)

非结构化

RGB (255, 0, 0)



⇒

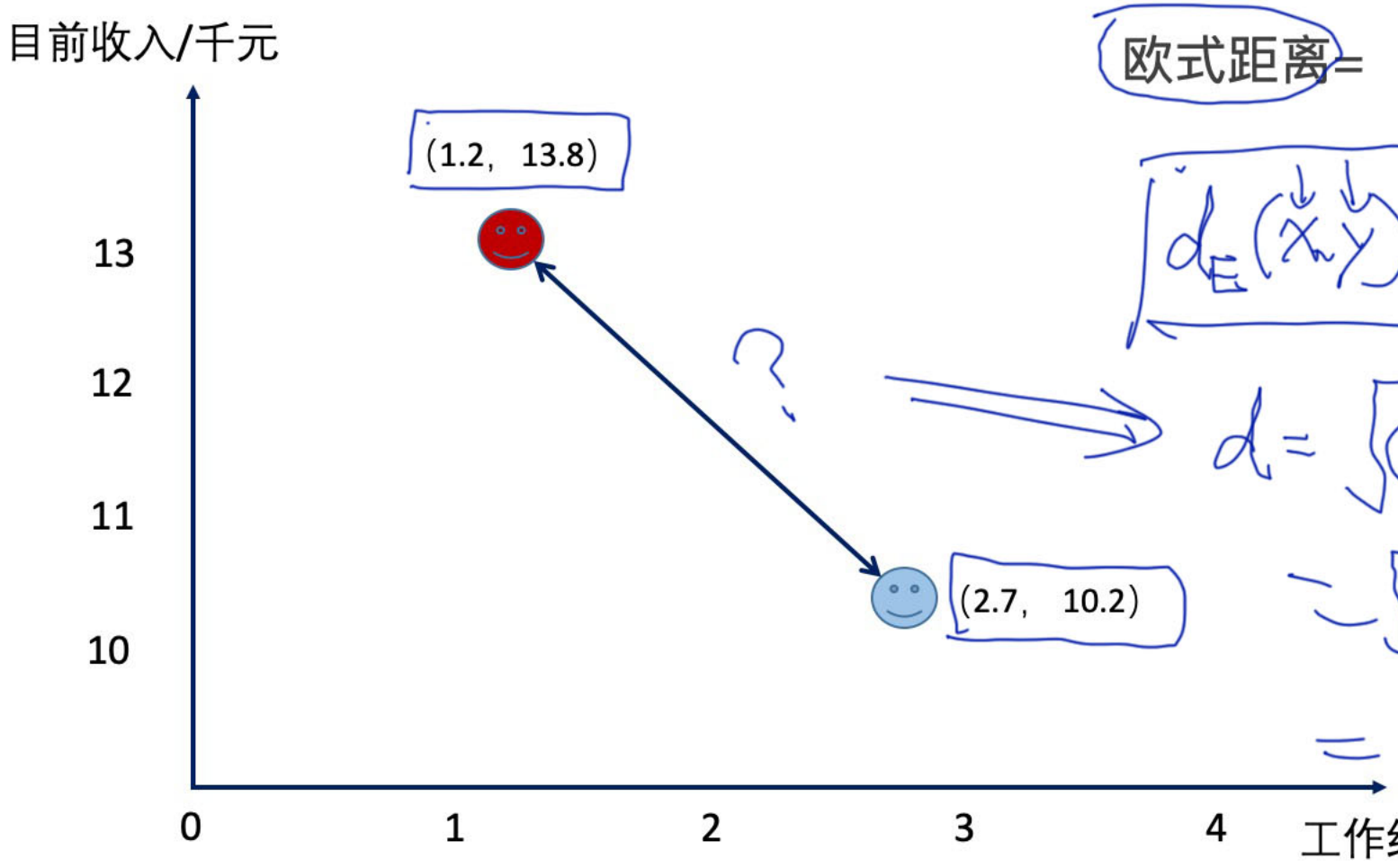
$$\begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix}$$

2. 标记号每个物体的标签

好人/坏人识别

水果种类识别

3. 计算两个物体之间的距离/相似度



欧式距离=

$x=(x_1, x_2, x_3)$ 3个特征/3维向

$y=(y_1, y_2, y_3)$

$$d_E(x, y)=\sqrt{(x_1-y_1)^2+(x_2-y_2)^2+(x_3-y_3)^2}$$

?

$$d=\sqrt{(1.2-2.7)^2+(13.8-10.2)^2}$$

$$=\sqrt{1.5^2+3.6^2}$$

$$=\sqrt{\ }\ ?$$

Coding Time–2

4. 选择合适的K

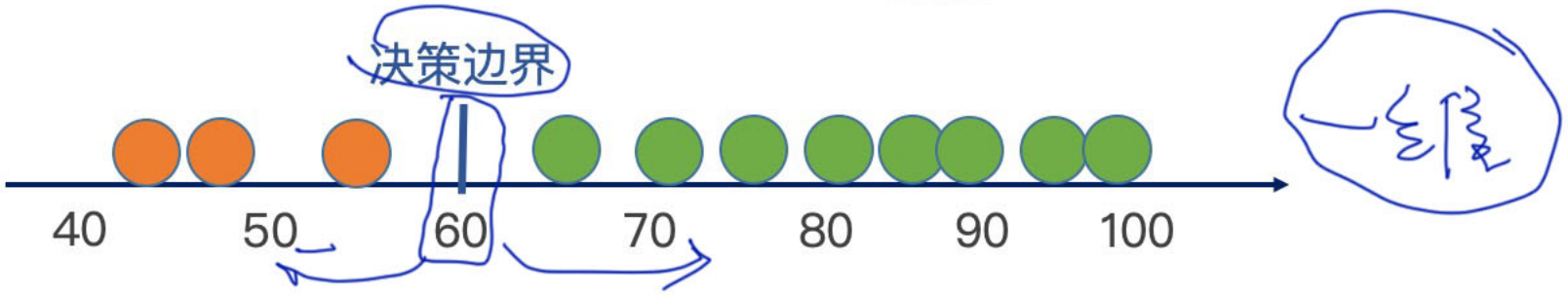
为了选择合理的K，首先需要去理解**K对算法的影响**

↓

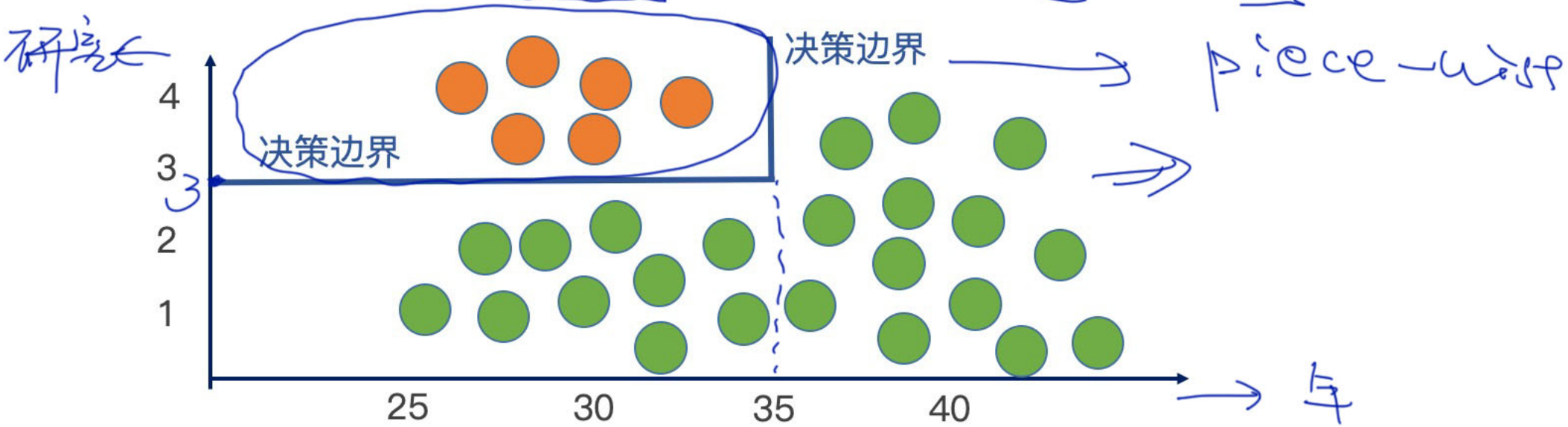
为了理解K对算法的影响，需要先理解什么叫算法的**决策边界**

决策边界

例1： 大学里60分以上作为及格，60分以下作为不及格 ← 评分（决策边）



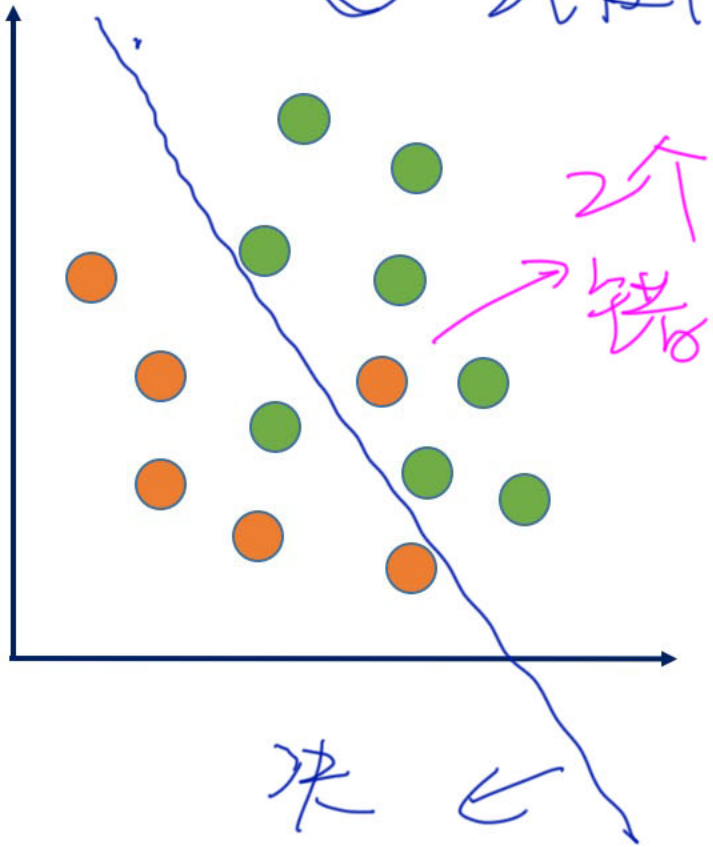
例2：今年某高校要计划引入35岁以下，具有海外研究经验3年以上的学者。



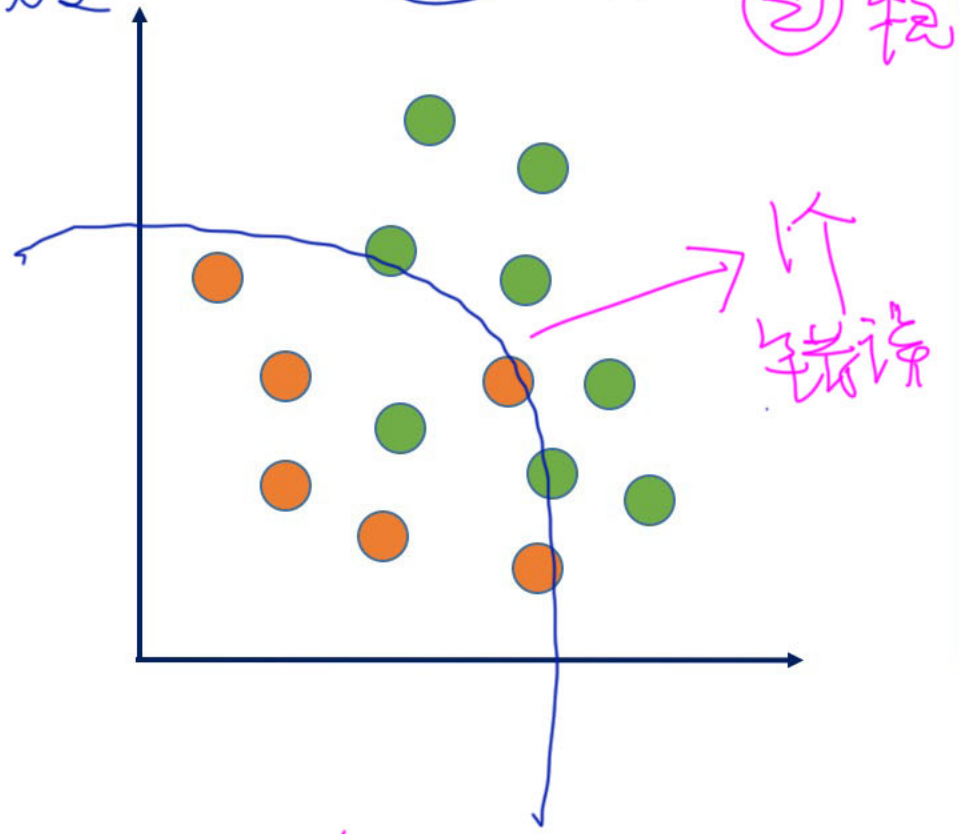
决策边界决定“线性分类器”或者“非线性分类器”

二分类

① 线性模型

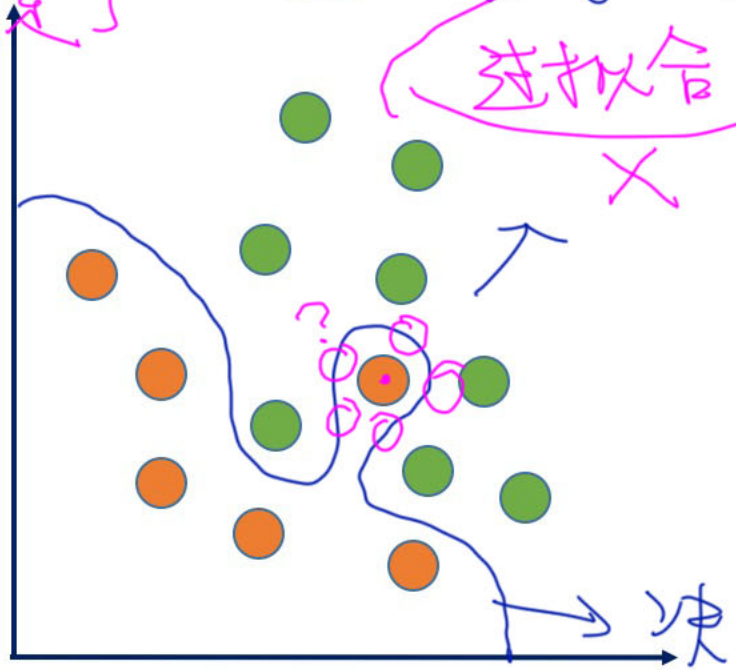


② 非线 ① 性价比 ② 稳定



③ 非线

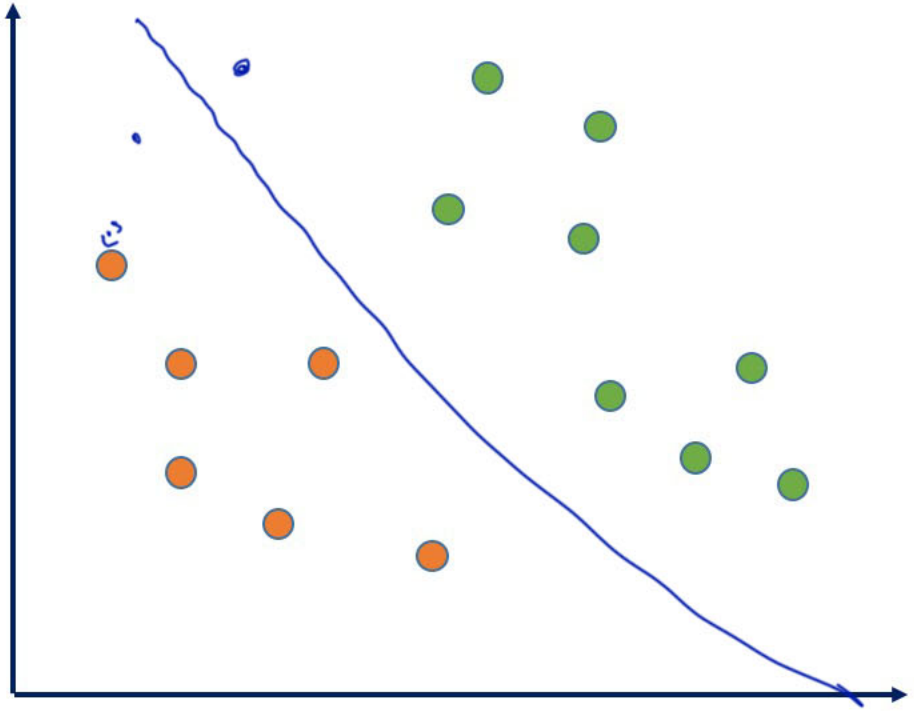
过拟合 ×



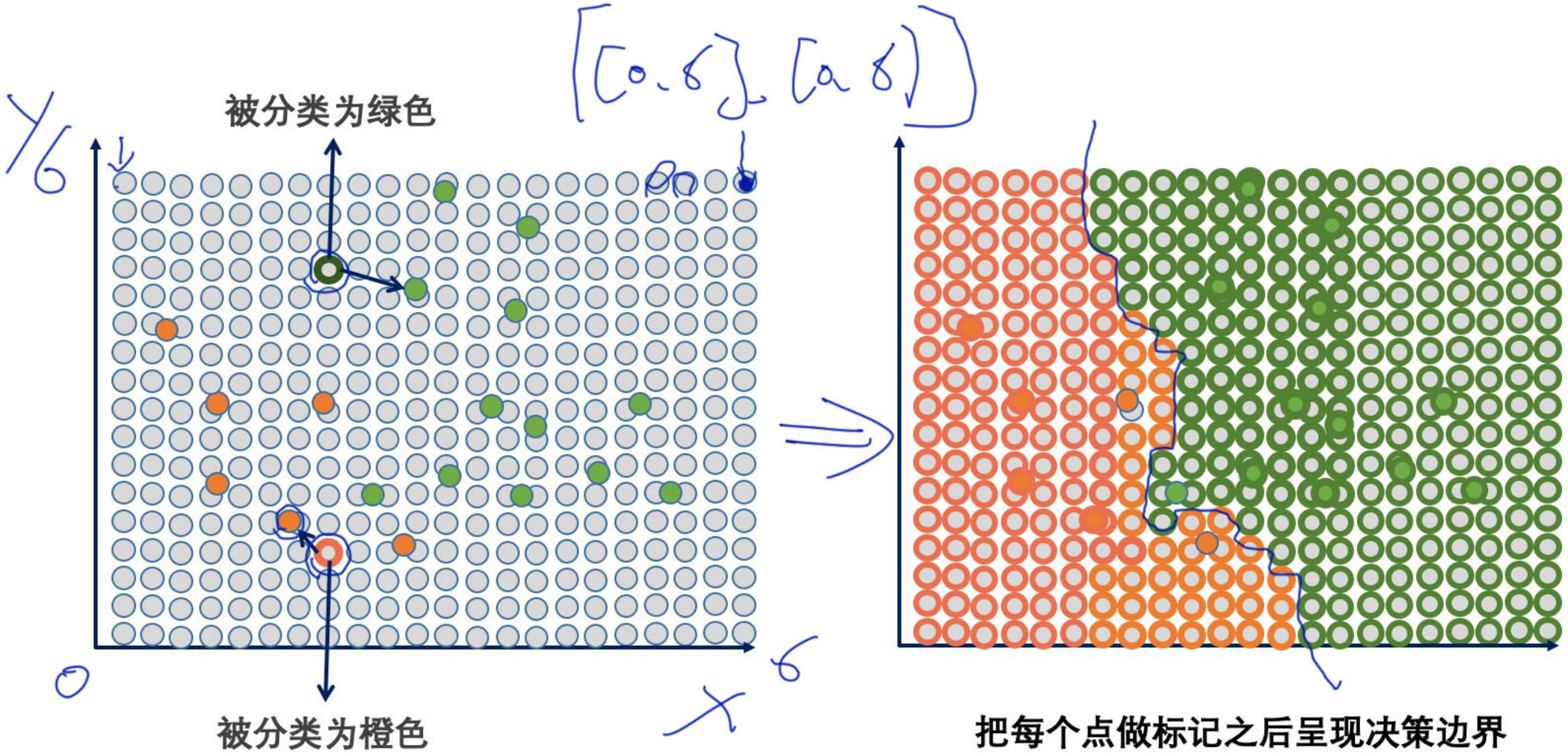
Which one is Best?

KNN的决策边界：怎么寻找（i.e., K=1）？

画？



KNN的决策边界：怎么寻找（i.e., K=1）？



把每个点做标记之后呈现决策边界

KNN的决策边界：K值的影响

问题：随着K的增加，会怎么变化?

→ 变得平滑 (smooth)

怎么去选择合适的K?

答案是**交叉验证（Cross Validation）** (CV)

通常，我们也把它归类为“调参”

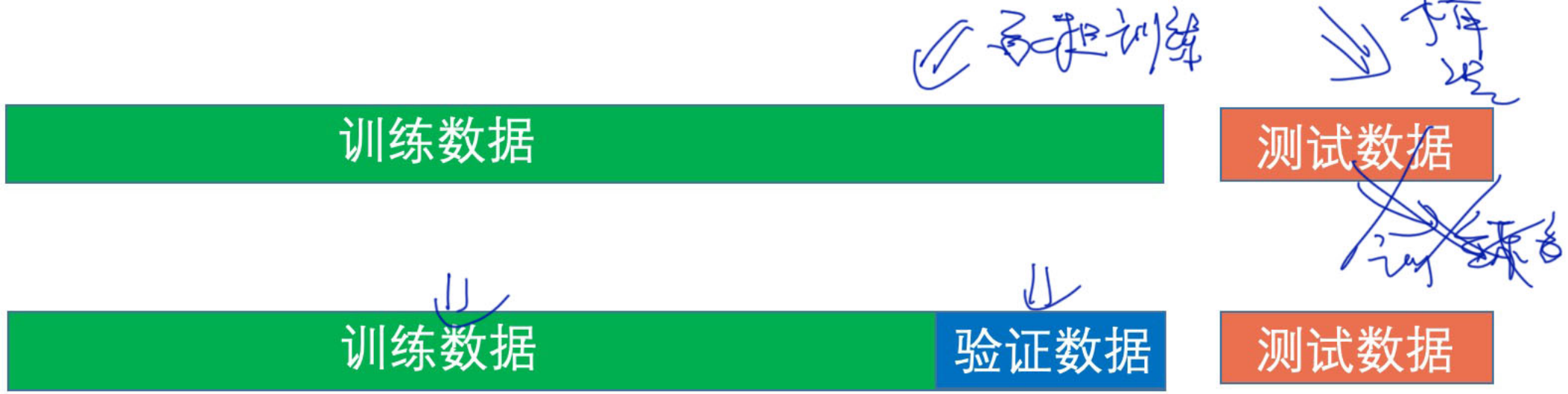
但什么是交叉验证？What? How?

交叉验证

k → k=1 → 作 k=2 → 作 → k=3

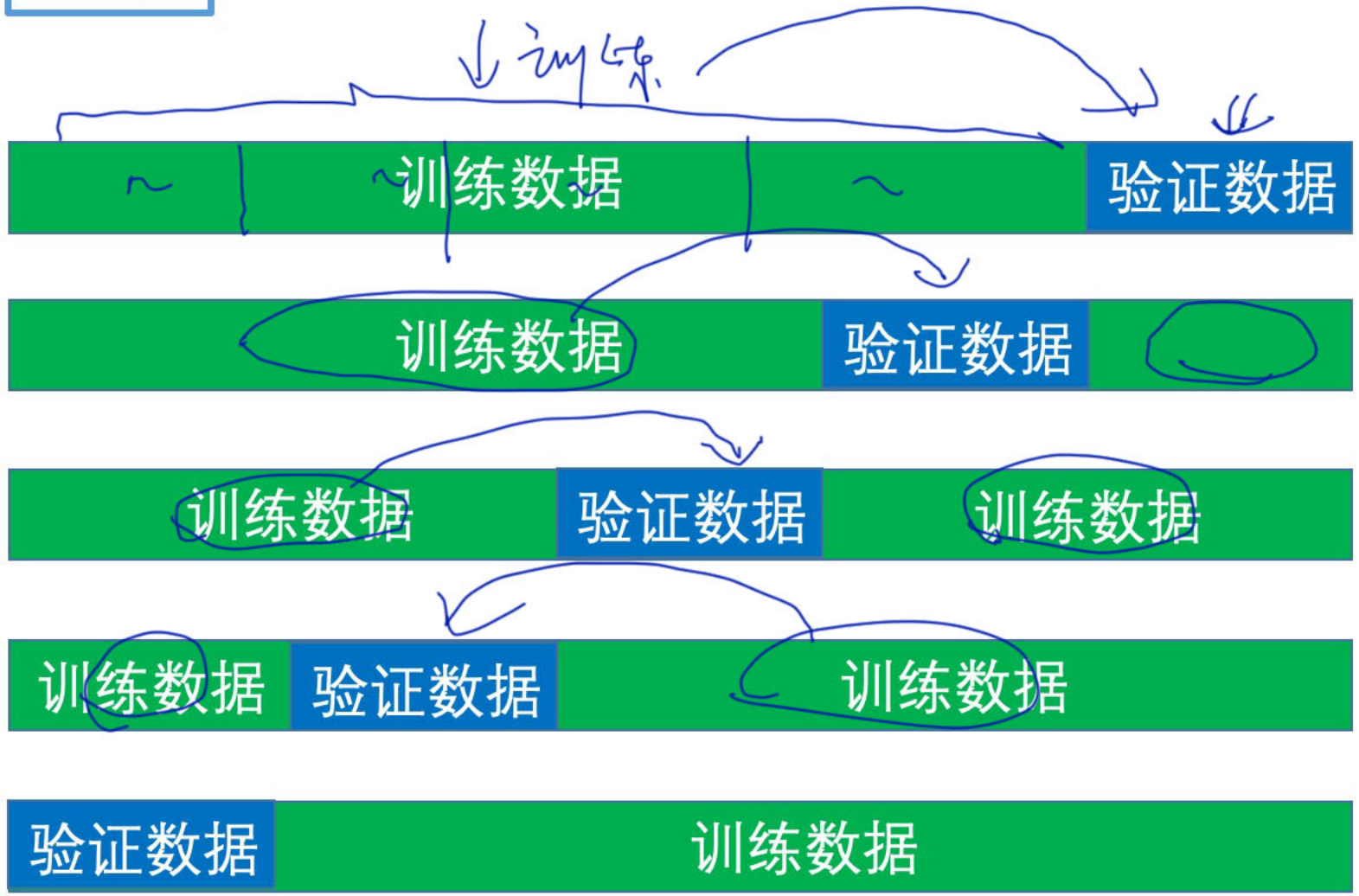
Intuition

把训练数据进一步分成训练数据（Training Data）和验证集(Validation Data)。 选择在验证数据里最好的超参数组合。



5折交叉验证(5-fold Cross Validation)

K=1 时：



平均

测试数据

acc_1

acc_2

acc_3

acc_4

acc_5

$$acc_{k=1} = \frac{(acc_1 + acc_2 + acc_3 + acc_4 + acc_5)}{5}$$

5折交叉验证(5-fold Cross Validation)

K=3 时：

k-fold cross

测试数据

acc_1

acc_2

acc_3

acc_4

acc_5

$$\overline{acc}_{k=3} = \frac{acc_1 + acc_2 + \cdots + acc_5}{5}$$

if $\overline{acc}_{k=1} > \overline{acc}_{k=3}$

$\boxed{k=1}$

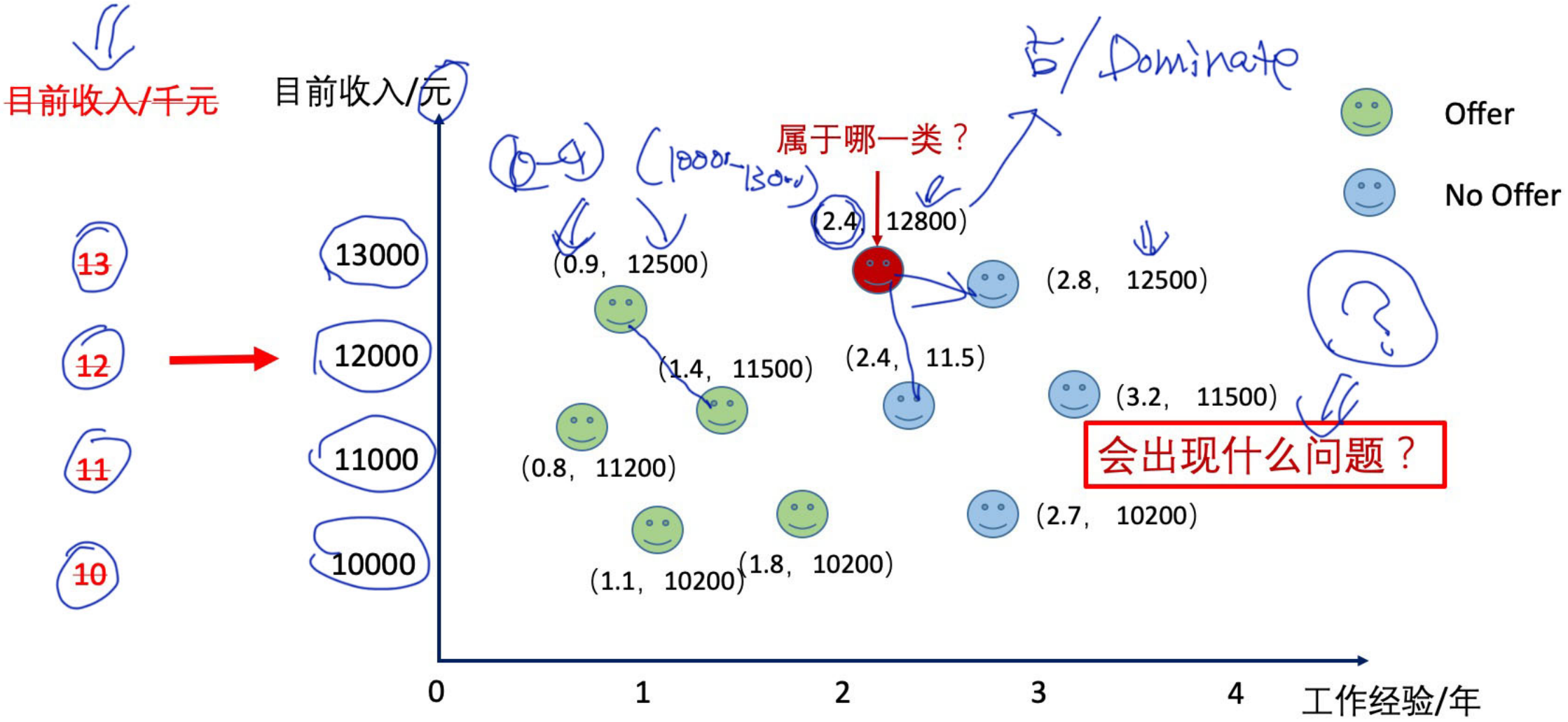
交叉验证中需要注意的点

1. 千万不能用测试数据来调参
2. 千万不能用测试数据来调参
3. 千万不能用测试数据来调参
4. 数据量越少，可以适当增加折数，why?

k 越大？

讨论题

KNN中需要考虑的点 — 回顾之前的例子



特征的缩放

(0, 60000)

[0, 10000] → [0, 1]

→ N(0, 1)

(0, 1)

1. 线性归一化（Min-max Normalization）

N(0, 1) ← 2. 标准差标准化（Z-score Normalization）

Normal Gaussian

特征缩放 — 线性归一化（Min–max Normalization）

OLD

特征
1.5
2
3
4
1.5
2.5
2
2.3
3
1.5

New

特征
0
0.2
0.6
1.0
0
0.4
0.2
0.33
0.6
0

数值范围 = [1.5，4]　　数值范围 = [0，1]

$$X_{new} = \frac{X - \min(X)}{\max(X) - \min(X)}$$

X的最小值

X的最大值

$\min(x) = 1.5$

$\max(x) = 4$

$= \frac{X - 1.5}{2.5}$

特征缩放 — 标准差标准化（Z-score Normalization）

N(0,1)

特征
1.5
2
3
4
1.5
2.5
2
2.3
3
1.5

特征
0
0.2
0.6
1.0
0
0.4
0.2
0.33
0.6
0

$$\frac{1.5 - mean(x)}{std(x)}$$

$$\frac{2 - mean(x)}{std(x)}$$

$$X_{new} = \frac{X - mean(X)}{std(X)}$$

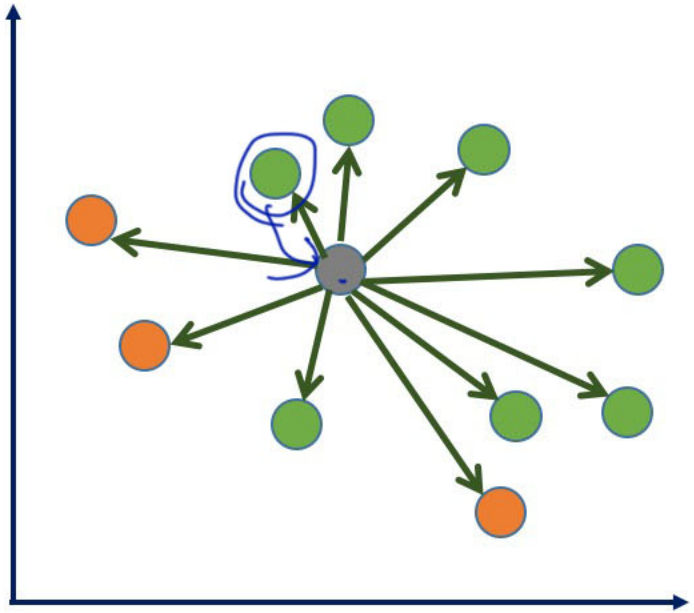
X的平均值

X的标准差

高斯

KNN 的延伸内容(1) — 如何处理大数据量?

$N = 10^6$



对新样本的分类

- 预测阶段：需要**计算被预测样本和每一个训练样本之间的距离**
- 时间复杂度：**O(N)**， N是样本总数
- 当N很大的时候，这个复杂度显然不能作为实时预测 ?

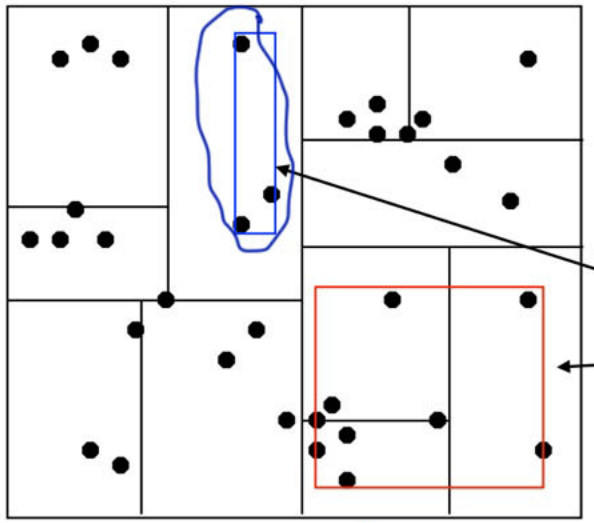
预测

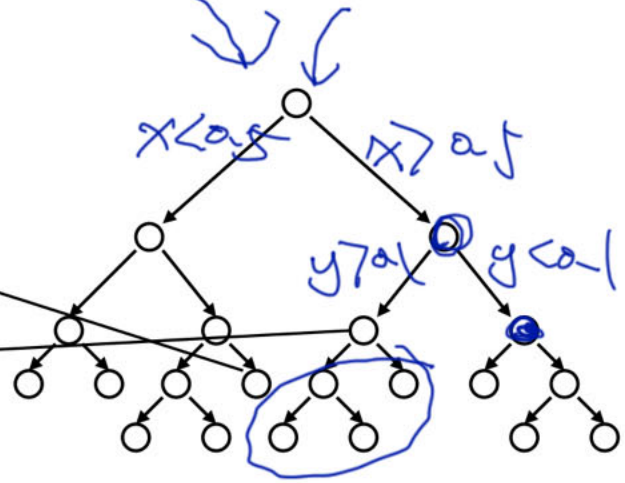
需要怎么办？

KNN：时间复杂度

KNN 的延伸内容(1) — 如何处理大数据量?

KD-tree





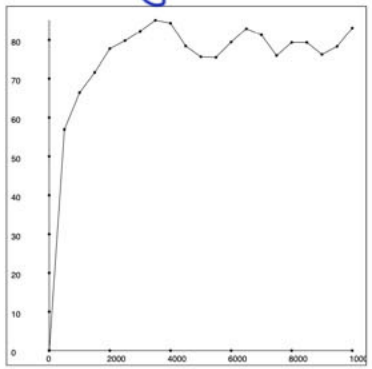
KD-tree

Geometric Properties

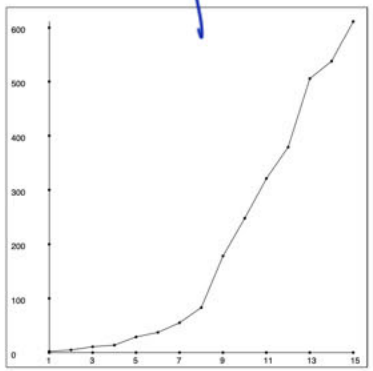
- 对于样本个数N来说O(logN)
- 对于数据维度d来说，指数级复杂度。

logN ✓

exponential



N



d

KD-tree 只能用在d小的

KNN 的延伸内容(1) – 如何处理大数据量?

近似算法：不再寻求完全准确的解，可以适当损失精确率

Stanford Large Scale --

利用类似**哈希**算法 – Locality Sensitivity Hashing (LSH)

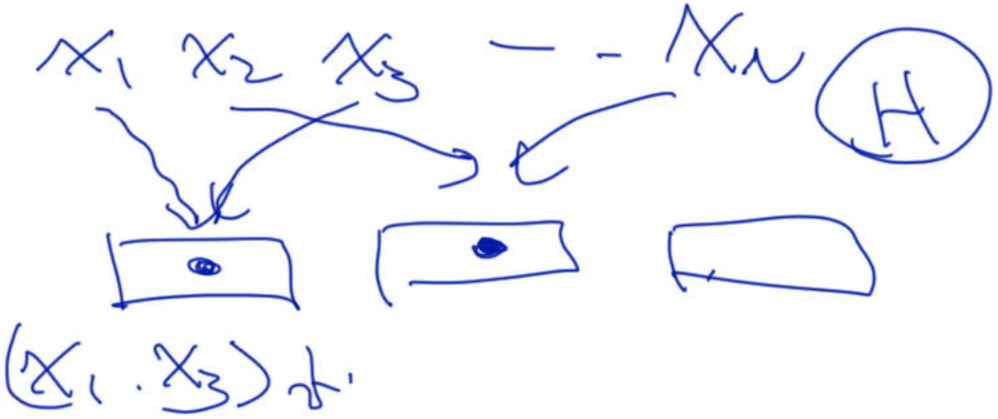
哈希

近似度

核心思想：把样本分在不同的bucket, 使得距离比较近的样本较大概率在同一个bucket里

LSH 函数h对于给定的相似度量函数d, 满足：

- d(x1, x2) <=r 时 p(h(x1)==h(x2))的概率高
- d(x1,x2)>αr时，p(h(x1)==h(x2))的概率低

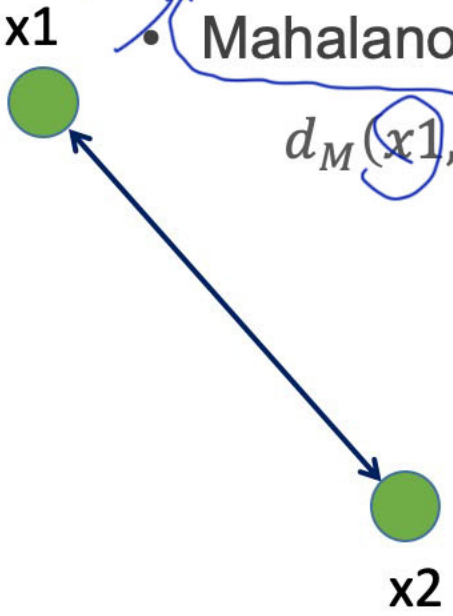


KNN 的延伸内容(2) — 如何处理特征之间的相关性?

$\sqrt{(x_{11}-x_{21})^2+(x_{12}-x_{22})^2 \cdots}$

$x_1=(x_{11}, x_{12}, x_{13})$

方差



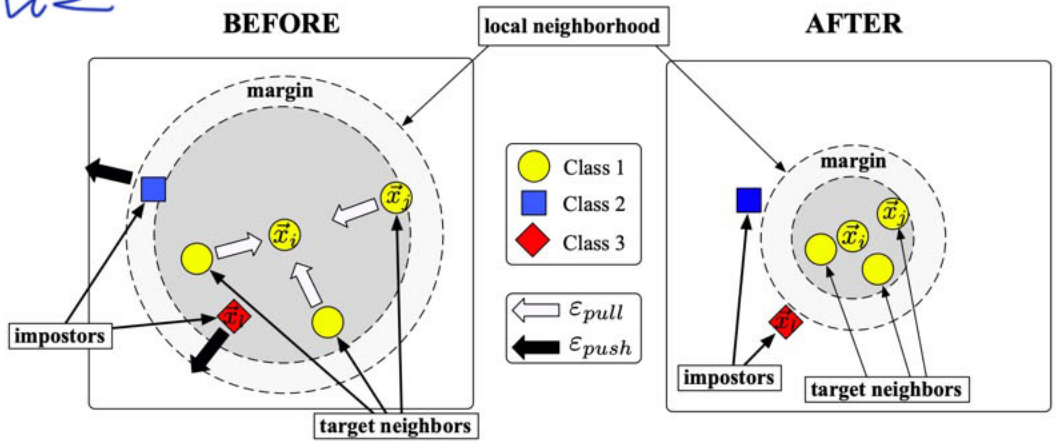
- Mahalanobis Distance

$$d_M(x1, x2) = (x1 - x2)^T M (x1 - x2)$$

I

参数矩阵M怎么去学习?

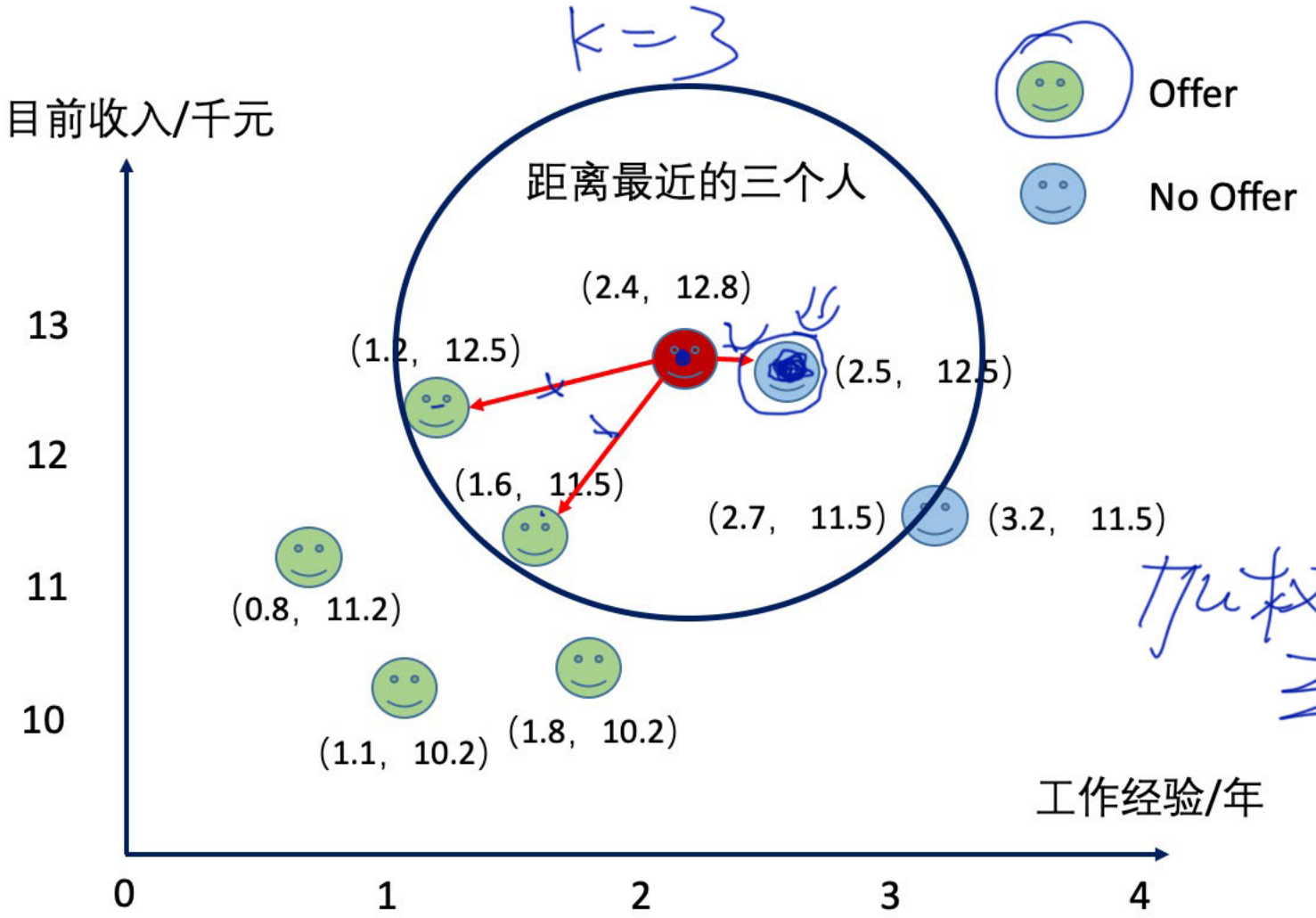
Large Margin Loss



参考: Kilian et al. 2009

LMNN

KNN 的延伸内容(3) — 怎么处理样本的重要性?



加权平均

Weighted KNN

权重计算

$$w(\mathbf{x}, \mathbf{x}_i) = \exp\left(-\lambda |\mathbf{x} - \mathbf{x}_i|_2^2\right)$$

预测

$$\Pr(y|\mathbf{x}) = \frac{\sum_{i=1}^{n} w(\mathbf{x}, \mathbf{x}_i)\delta(y, y_i)}{\sum_{i=1}^{n} w(\mathbf{x}, \mathbf{x}_i)}$$

$$\delta(y, y_i) = \begin{cases} 1 & y = y_i \\ 0 & y \neq y_i \end{cases}$$

KNN 的延伸内容(4) — 能不能利用Kernel Trick?

强调 相似度

SVM → **Kernelized KNN**

Epanechnikov quadratic kernel

$$K_\lambda(x_0,x)=D\left(\frac{|x-x_0|}{\lambda}\right) \quad D(t)=\begin{cases}\frac{3}{4}(1-t^2) & \text{if } |t|\le 1;\\ 0 & \text{otherwise.}\end{cases}$$

λ = bandwidth

tri-cube kernel

$$K_\lambda(x_0,x)=D\left(\frac{|x-x_0|}{\lambda}\right) \quad D(t)=\begin{cases}(1-|t|^3)^3 & \text{if } |t|\le 1;\\ 0 & \text{otherw}\end{cases}$$

Gaussian kernel

$$K_\lambda(x_0,x)=\frac{1}{\sqrt{2\pi}\lambda}\exp\left(-\frac{(x-x_0)^2}{2\lambda^2}\right)$$

总结

1. KNN是一个非常简单的算法
2. 比较适合应用在低维空间
3. 预测时候的复杂度高，对于大数据需要一定的处理

kd-tree LSH

$D =$

特征 Label/标签

$x^{(1)} = (0.3, 0.7, 0.6)$

$x^{(2)} = (0.5, 0.6, 0.5)$

$x^{(3)} = (0.6, 0.7, 1.1)$

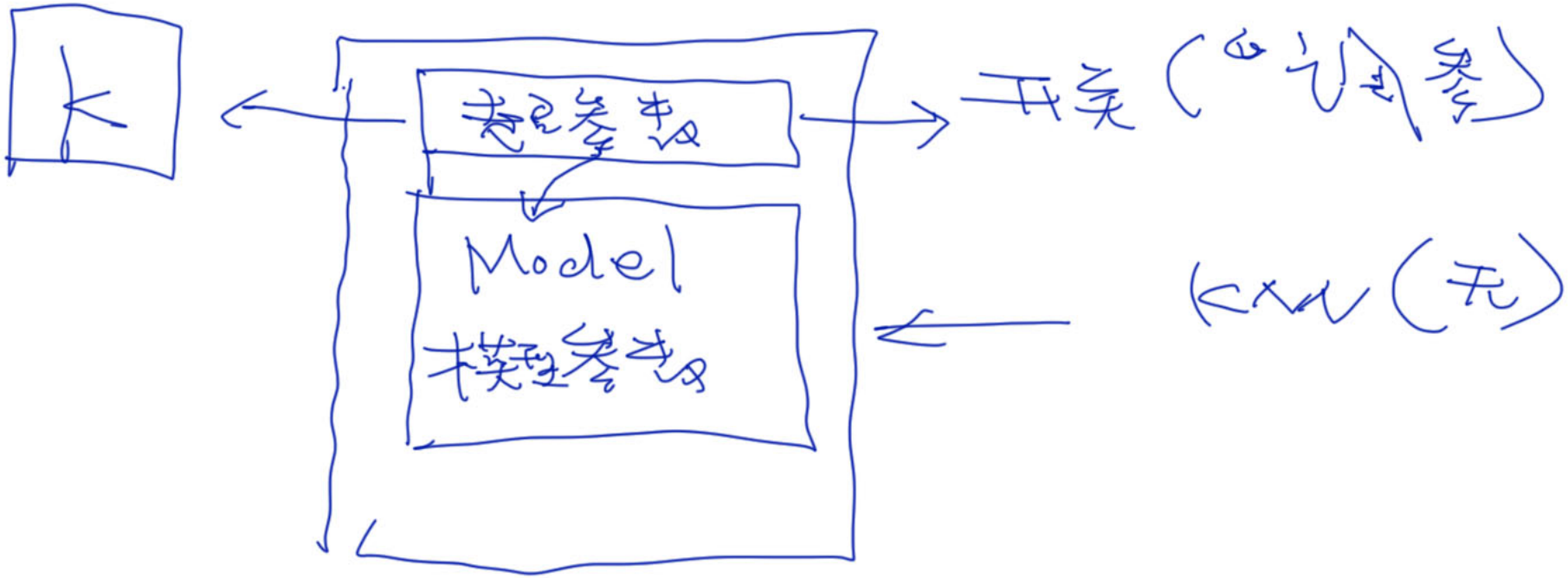
$x^{(4)} = (0.8, 0.5, 1.2)$

x^{New}

Q: KNN 怎么用在回归问题上?

$k=3$

$$y^{New} = \frac{0.5+1.1+1.2}{3}$$

参数 → ① 模型参数(Model Parameter) ← 数据

参数 → ② 超参数(Hyperparameter) ← 不属于