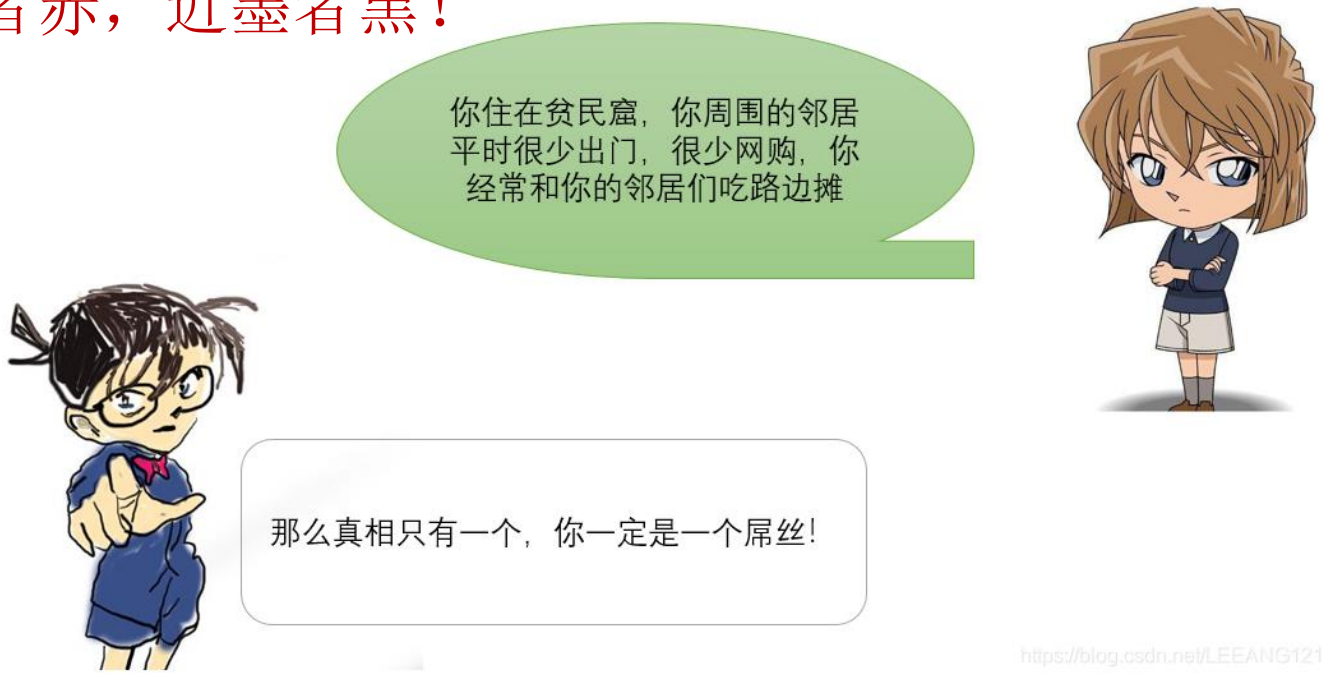
K-近邻算法

K–Nearest Neighbor

KNN

- k近邻（k Nearest Neighbor, kNN）的核心思想
  - 一种比较成熟也是较为简单的机器学习分类和回归算法
  - 其核心思想是：如果一个样本在特征空间中的k个最相邻的样本中的大多数属于某一个类别，则该样本也属于这个类别，并具有这个类别上样本的特性。
  - 近朱者赤，近墨者黑！



KNN

- k近邻（k Nearest Neighbor, kNN）的核心思想
  - 对于分类问题：对新的样本，根据其k个最近邻的训练样本的类别，通过多数表决等方式进行预测。
  - 对于回归问题：对新的样本，根据其k个最近邻的训练样本标签值的均值作为预测值。
  - k近邻法的三要素：

    •距离度量

    • k值选择

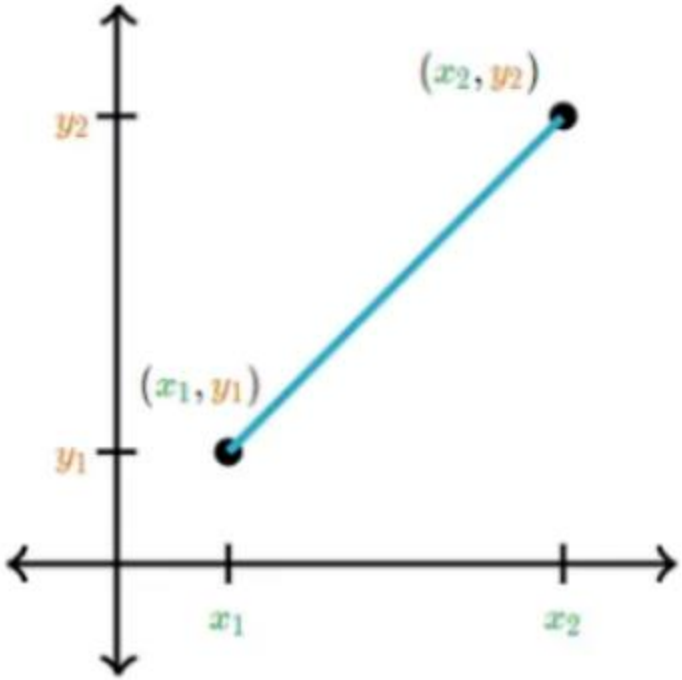
    •决策规则

KNN

- 距离的度量
  - 距离度量，简而言之，是一种衡量数据集中元素之间关系(相似性/差异性）的方法。
  - 它通过定义距离函数来实现，这个函数为数据集中的每个元素提供了一种相互关系的度量
  - 距离函数，本质上，是一种数学工具，它帮助我们量化数据集中任意两个元素之间的差异
  - 不同的距离度量采用不同的数学公式作为其距离函数
  - 如果两个元素之间的距离为零，可以认为它们是等同的
  - 如果距离大于零，则它们有所不同

KNN

- 距离的度量
  - 欧氏距离(Euclidean distance)

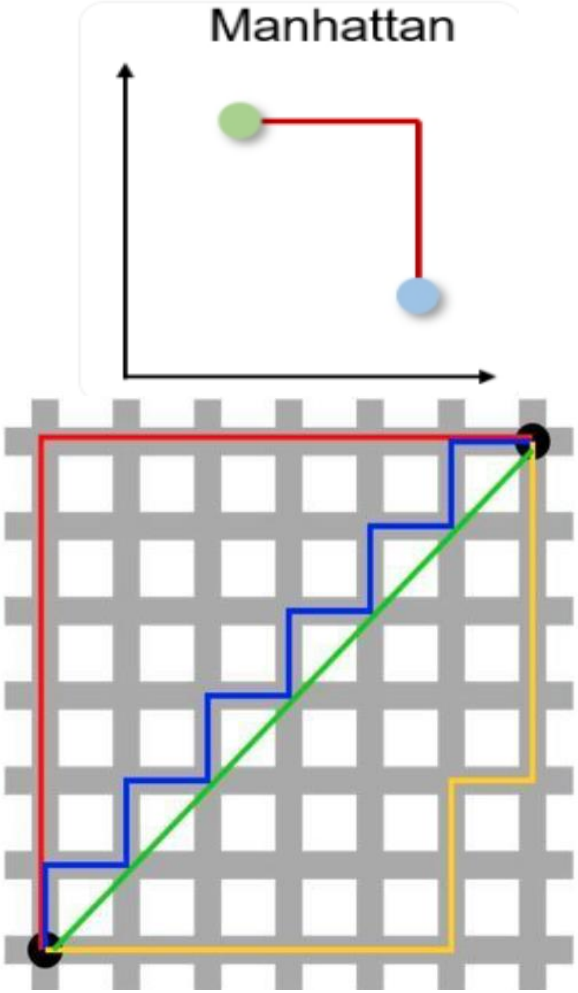


$$d(x,y) = \sqrt{\sum_i (x_i - y_i)^2}$$

欧几里得度量（也称欧氏距离）是一个通常采用的距离定义，指在m维空间中两个点之间的真实距离，或者向量的自然长度（即该点到原点的距离）。在二维和三维空间中的欧氏距离就是两点之间的实际距离

KNN

- 距离的度量
  - 曼哈顿距离(Manhattan distance)

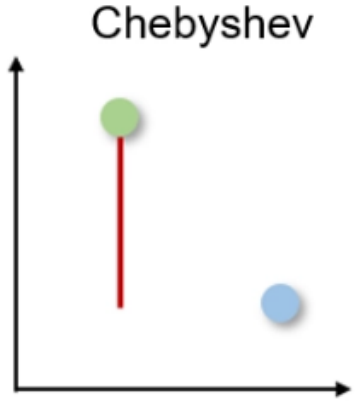


$$d(x,y) = \sum_{i} |x_i - y_i|$$

- 曼哈顿距离被称为出租车或城市街区距离，两个实值向量之间的距离是根据一个人只能以直角移动计算的。想象你在城市道路里，要从一个十字路口开车到另外一个十字路口，实际驾驶距离就是曼哈顿距离
- 这种距离度量通常用于需要在网格状路径中计算距离的场景，如城市街区或棋盘，适合离散和二元属性，这样可以获得真实的路径

KNN

- 距离的度量
  - 切比雪夫距离(Chebyshev distance)



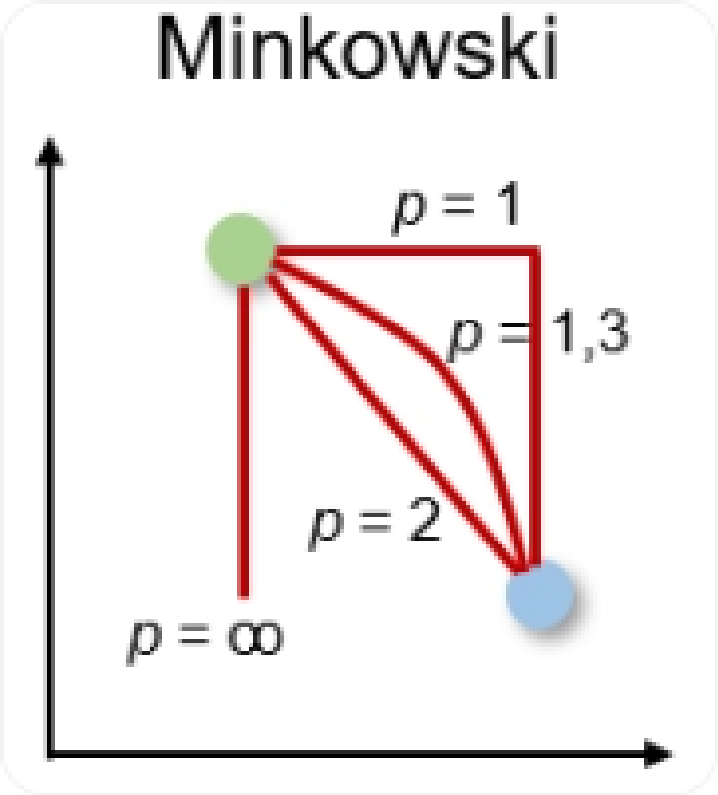
	a	b	c	d	e	f	g	h	
8	5	4	3	2	2	2	2	2	8
7	5	4	3	2	1	1	1	2	7
6	5	4	3	2	1	♔	1	2	6
5	5	4	3	2	1	1	1	2	5
4	5	4	3	2	2	2	2	2	4
3	5	4	3	3	3	3	3	3	3
2	5	4	4	4	4	4	4	4	2
1	5	5	5	5	5	5	5	5	1
	a	b	c	d	e	f	g	h	

$$d(x,y) = \max_i |x_i - y_i|$$

- 切比雪夫距离也称为棋盘距离，它是两个实值向量之间任意维度上的最大距离。
- 国际象棋棋盘上二个位置间的切比雪夫距离是指王要从一个位子移至另一个位子需要走的步数。由于王可以往斜前或斜后方向移动一格，因此可以较有效率的到达目的的格子。上图是棋盘上所有位置距f6位置的切比雪夫距离

KNN

- 距离的度量
  - 闵可夫斯基距离 （Minkowski distance)



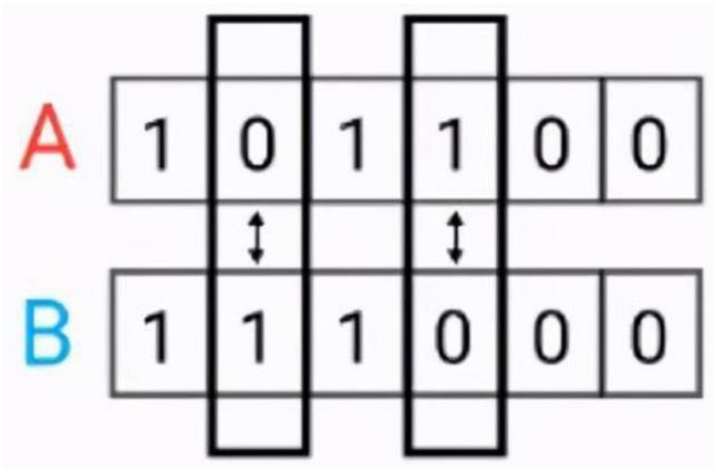
$$d(x,y) = \left(\sum_i |x_i - y_i|^p \right)^{\frac{1}{p}}$$

- 闵可夫斯基距离是一种在范数向量空间中使用的度量
- 闵可夫斯基距离是前述距离度量的广义形式
- p取2即为欧氏距离，
- p取1时则为曼哈顿距离。
- 当p 取无穷时的极限情况下，可以得到切比雪夫距离
- p取1或2时的距离是最为常用的

KNN

- 距离的度量
  - 汉明距离(Hamming distance)

$$d(x,y) = \frac{1}{N}\sum_{i} 1_{x_i \neq y_i}$$

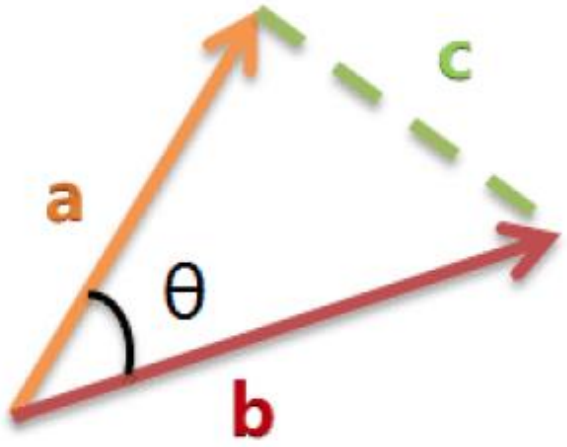


- 汉明距离衡量两个二进制向量或字符串之间的差异
- 汉明距离是使用在数据传输差错控制编码里面的，它表示两个相同长度的向量对应位不同的数量，我们以表示两个字之间的汉明距离。对两个字符串进行异或运算，并统计结果为1的个数，那么这个数就是汉明距离。

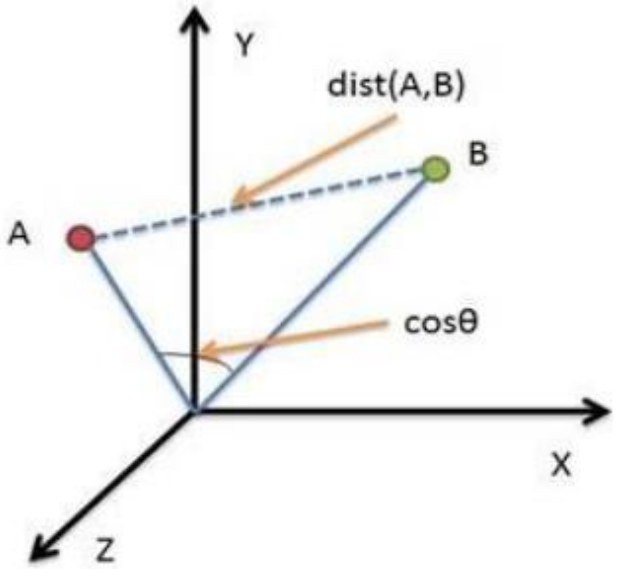
KNN

- 距离的度量
  - 余弦相似度

$$\cos(\theta) = \frac{A \cdot B}{\| A \| \| B \|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2} \times \sqrt{\sum_{i=1}^{n} (B_i)^2}}$$



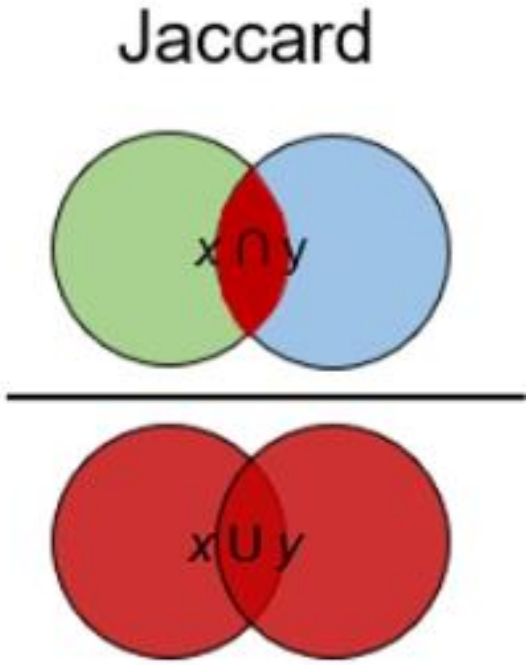
- 余弦相似度是方向的度量，他的大小由两个向量之间的余弦决定，并且忽略了向量的大小。
- 余弦相似度通常用于与数据大小无关紧要的高维空间中的问题，如推荐系统或文本分析中。
- 两个向量有相同的指向时，余弦相似度的值为1；两个向量夹角为90度时，余弦相似度的值为0；两个向量指向完全相反的方向时，余弦相似度的值为 1



KNN

- 距离的度量
  - 杰卡德系数（Jaccard Index）和距离

$$d = 1 - \frac{|x \cap y|}{|x \cup y|}$$



- Jaccard系数用于确定两个样本集之间的相似性。
- 它反映了与整个数据集相比存在多少一对一匹配。
- Jaccard系数通常用于二进制数据比如图像识别的深度学习模型的预测与标记数据进行比较，或者根据单词的重叠来比较文档中的文本模式

KNN

- 算法流程如下：

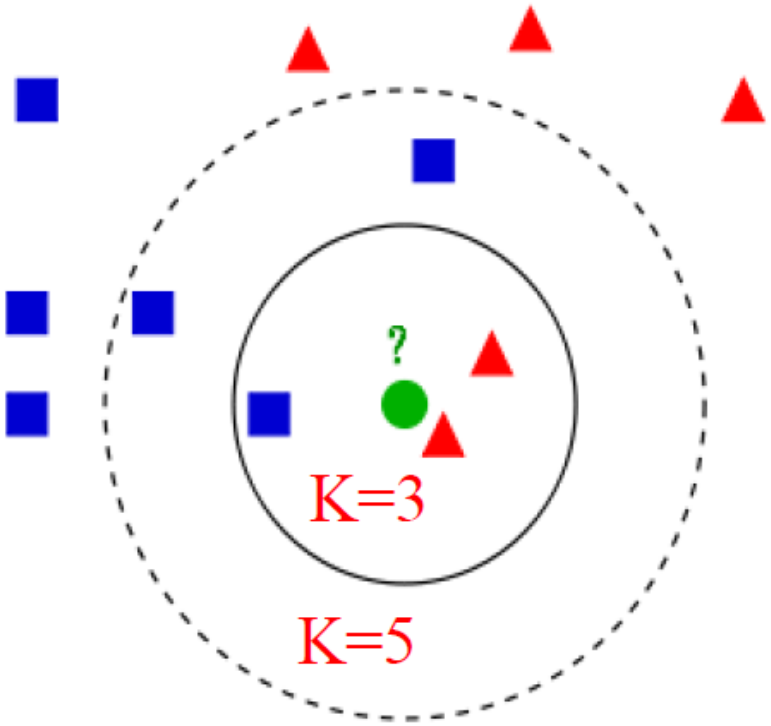
1、计算测试对象到训练集中每个对象的距离

2、按照距离的远近排序

3、选取与当前测试对象最近的k 的训练对象，作为该测试对象的邻居

4、统计这k 个邻居的类别频次

5、k 个邻居里频次最高的类别，即为测试对象的类别

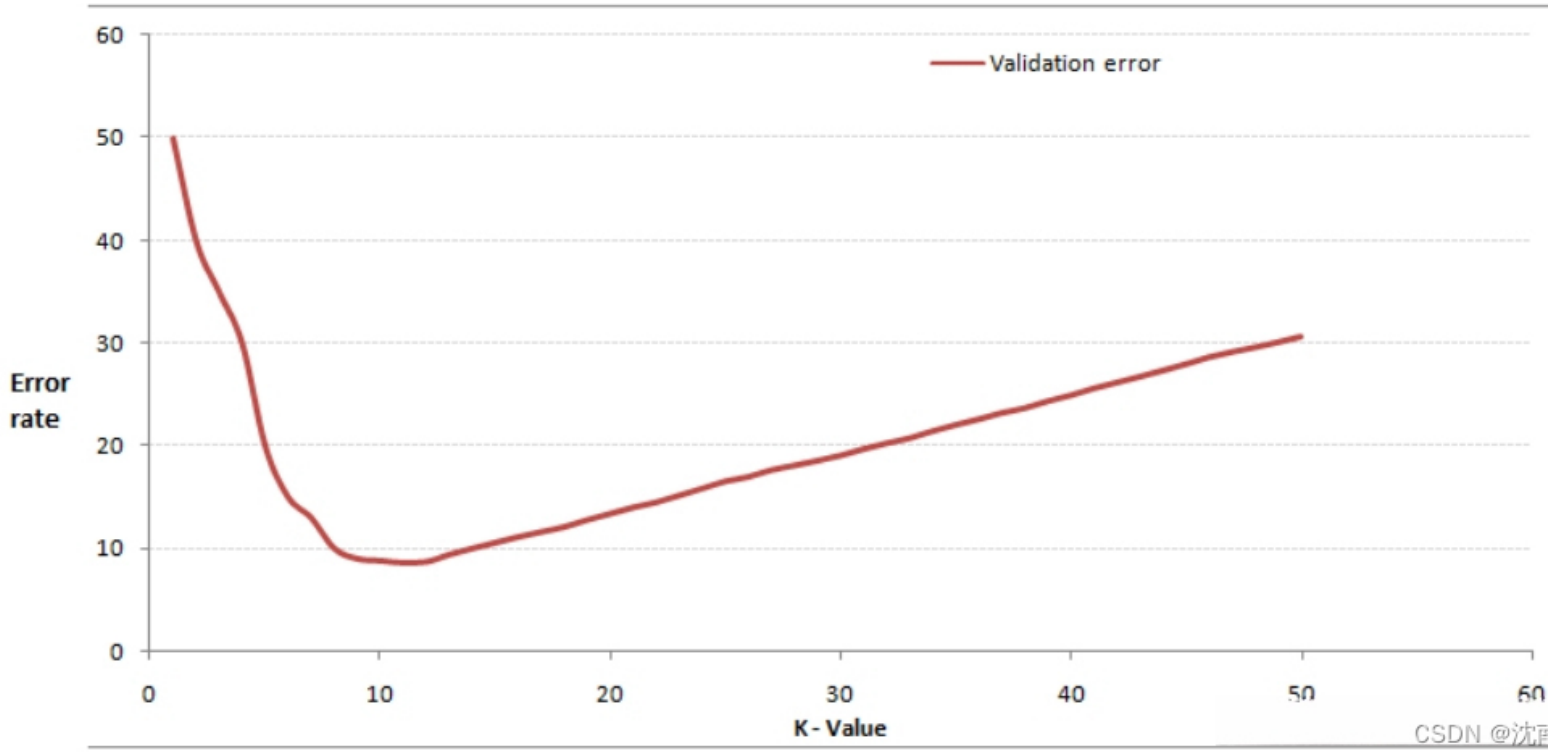


KNN

- K值选取
  - K值并不是越小越好也不是越大越好，而是差不多才好
  - 如果K值太小，分类的错误率会比较高，容易过拟合
  - 如果K值太大，那么分类就会一直倾向其中数量多的某一类
  - 所以，K值取一个合适的值是最重要的

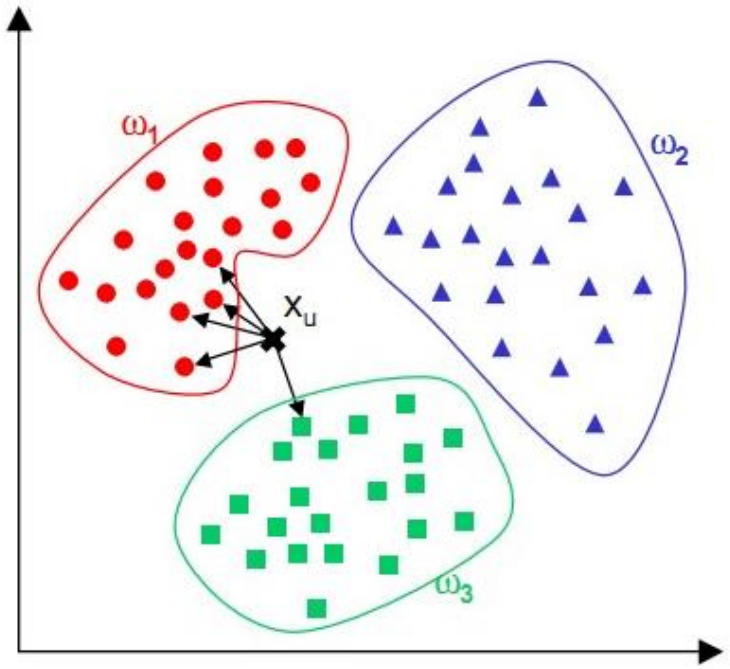
KNN

- K值选取
  - 使用交叉验证来确定K值选取
  - 在给定的数据集中，拿出大部分比例的数据进行模型的构建，保留小部分的数据进行模型的验证，来对错误率进行记录，分析得到最佳的K值



KNN

- 决策准则
  - 多数表决：对于分类问题中新的样本，根据其k个最近邻的训练样本的类别，通过多数表决等方式进行预测。
  - 平均法：对于回归问题中新的样本，根据其k个最近邻的训练样本标签值的均值作为预测值。



KNN

- 例子
  - 电影分类，已知各种电影的分类和相关的特征，求《唐人街探案》这部电影的分类。

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

《唐人街探案》VS《伦敦陷落》

$$d = \sqrt{(23-2)^2 + (3-3)^2 + (17-55)^2} = 43.42$$

序号	电影名称	搞笑镜头	拥抱镜头	打斗镜头	电影类型
1	功夫熊猫	39	0	31	喜剧片
2	叶问3	3	2	65	动作片
3	伦敦陷落	2	3	55	动作片
4	代理情人	9	38	2	爱情片
5	新步步惊心	8	34	17	爱情片
6	谍影重重	5	2	57	动作片
7	功夫熊猫	39	0	31	喜剧片
8	美人鱼	21	17	5	喜剧片
9	宝贝当家	45	2	9	喜剧片
10	唐人街探案	23	3	17	?

KNN

- 例子
  - 电影分类，已知各种电影的分类和相关的特征，求《唐人街探案》这部电影的分类。

序号	电影名称	搞笑镜头	拥抱镜头	打斗镜头	电影类型	距离	K=5时
1	功夫熊猫	39	0	31	喜剧片	21.47	√
2	叶问3	3	2	65	动作片	52.01	
3	伦敦陷落	2	3	55	动作片	43.42	
4	代理情人	9	38	2	爱情片	40.57	
5	新步步惊心	8	34	17	爱情片	34.44	√
6	谍影重重	5	2	57	动作片	43.87	
7	功夫熊猫	39	0	31	喜剧片	21.47	√
8	美人鱼	21	17	5	喜剧片	18.55	√
9	宝贝当家	45	2	9	喜剧片	23.43	√
10	唐人街探案	23	3	17	?	——	?

喜剧片！

KNN

- 应用场景
  - 电影分类
  - 图像分类：识别人脸、车牌等。
  - 文本分类：垃圾邮件过滤、情感分析等。
  - 推荐系统：根据用户的历史行为推荐商品、音乐等。
  - 数据挖掘：寻找异常值、聚类等。
  - 生物信息学：基因分类、蛋白质分类等。
  - 财务分析：预测股票价格、评估信用风险等。

KNN总结

- 优点
  - 算法逻辑简单
  - 很少的训练，或者说不需要训练。
  - 不对数据分布作出假设，完全基于距离度量对样本特征进行提取
  - 预测效果较好，容错能力强。对于异常值的处理能力较强，不易受异常值的影响
  - 适用于多类别问题。既可以用于分类，也可以用于回归问题，且可以进行非线性分类

KNN总结

- 缺点
  - 内存要求高，在进行预测时，需要存储所有的训练数据在内存中
  - 计算成本高，当特征数较多时，计算的成本会很大
  - K值选取不易，对于上文所说的K值选取如果不当，会出现很大的误差
  - 样本不平衡问题，当某些类别的样本数量极不均匀时，KNN算法可能无法正确分类
  - 速度慢，KNN是一种消极学习方法，懒惰算法，导致预测时速度比起逻辑回归之类的算法慢