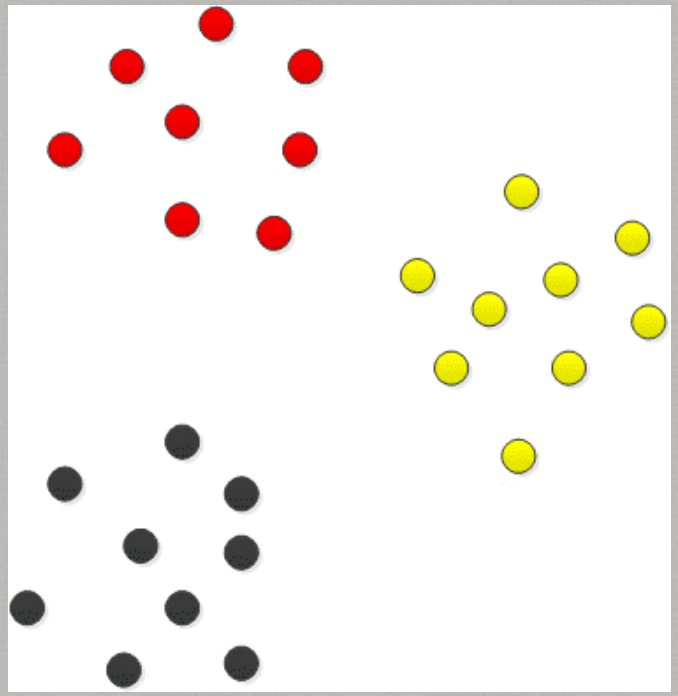
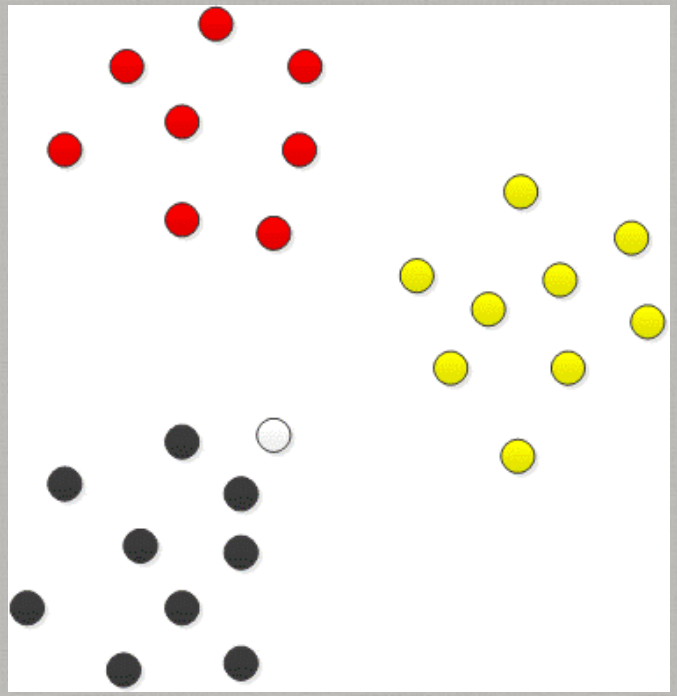
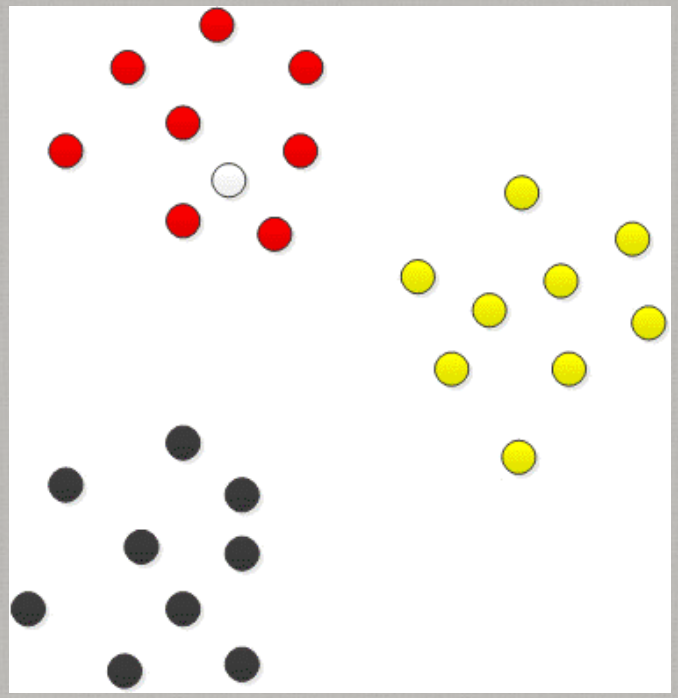
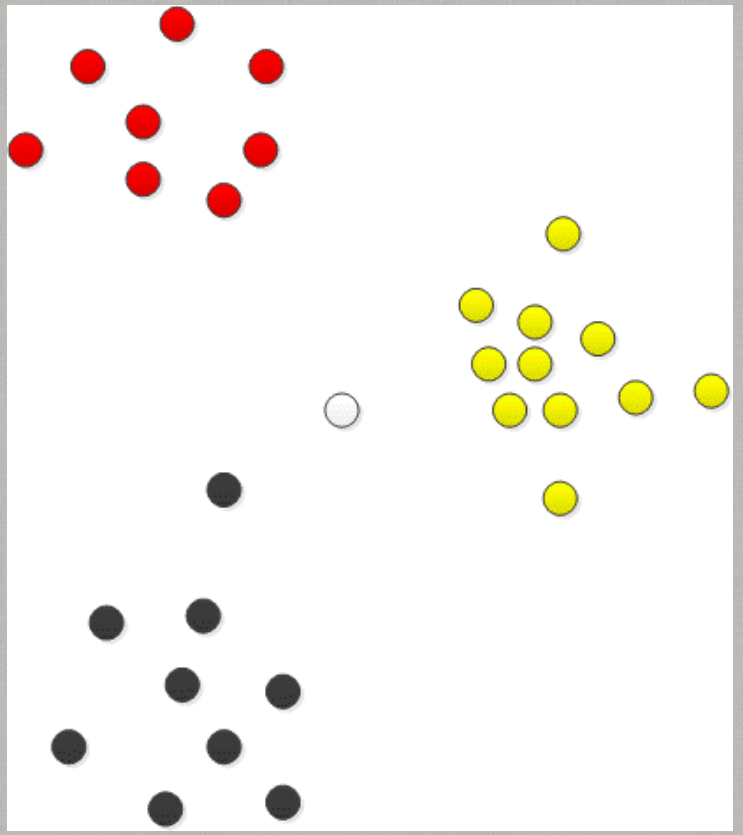
# K - Nearest Neighbor Method – An Instance based method

KNN

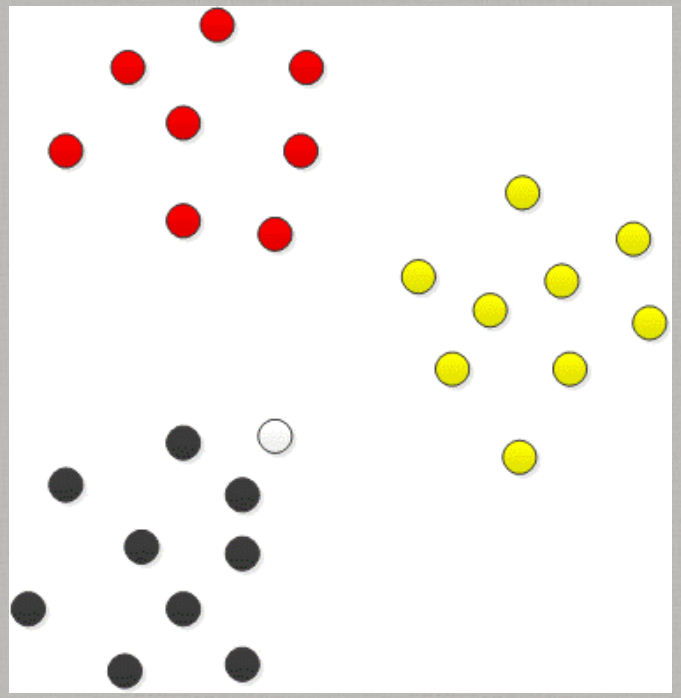






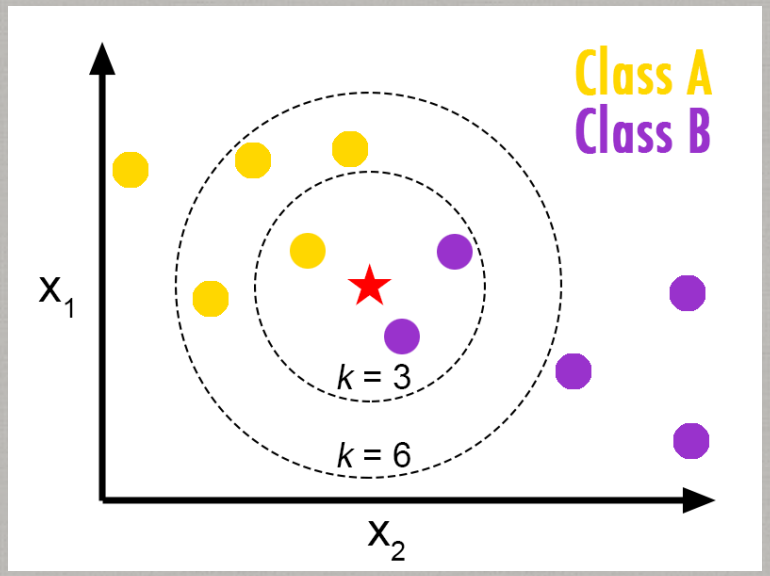


# The Nearest Neighbor Method



- ***Observation:*** similar examples tend to belong to the same class.
- ***Idea:*** assign a test input with the class label from the example in the training data set that is the most similar to the test input.
- ***Implementation:*** calculate the distances between the test input and the examples in the data set and find the one with the shortest distance.

# The K - Nearest Neighbor Method




- ***Observation:*** similar examples tend to belong to the same class.
- ***Idea:*** assign a test input with the dominating class label from the examples in the training data set that are within the neighborhood of the test input.
- ***Implementation:*** calculate the distances between the test input and the examples in the training data set and find the K ones that are the most closest to the test input and find the dominating class label within these K neighboring examples.

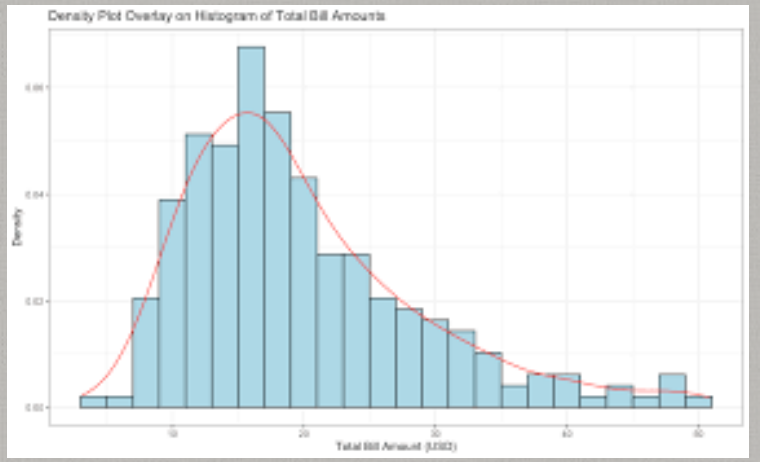
# ***Histogram*** density model

In the case of a single continuous variable $x$. Given $N$ sample values from $x$.

- Partition $x$ into distinct bins of width $\Delta_i$
- Count the number of observations of $x$ falling in bin $i$ as $n_i$
- Probability density value for bin $i$ is given by

$$p_i = \frac{n_i}{N\Delta_i}$$

Hence, we have, $\sum_{i=1}^{m} p_i \Delta_i = 1$



- Let's suppose we have a data set comprising $N_k$ points in class $\mathcal{C}_k$ with $N$ points in total so that $\sum_{k=1}^{C} N_k = N$.

- If we wish to classify a new point $\boldsymbol{x}$, we draw a sphere centered on $\boldsymbol{x}$ containing precisely $K$ points irrespective of their class. Suppose this sphere has volume $V$ and contains $K_k$ points from class $\mathcal{C}_k$.

- We want to classify the new point $\boldsymbol{x}$ to class $\mathcal{C}_k$ that has the highest ***posterior probability of class membership:***

$$p(\mathcal{C}_k|\boldsymbol{x}) = \frac{p(\boldsymbol{x}|\mathcal{C}_k)p(\mathcal{C}_k)}{p(\boldsymbol{x})}$$

- Then the likelihood can be estimated as:

$$p(\boldsymbol{x}|\mathcal{C}_k) = \frac{K_k}{N_k V}$$

- The unconditional density can be estimated as:

$$p(\boldsymbol{x}) = \frac{K}{NV}$$

- The class prior can be calculated as:

$$p(\mathcal{C}_k) = \frac{N_k}{N}$$

- Applying ***Bayes' theorem*** gives the ***posterior probability of class membership***:

$$p(\mathcal{C}_k|\boldsymbol{x}) = \frac{p(\boldsymbol{x}|\mathcal{C}_k)p(\mathcal{C}_k)}{p(\boldsymbol{x})} = \frac{K_k}{K}$$

- To minimize the probability of misclassification, we assign the test point $\boldsymbol{x}$ to the class having the largest posterior probability, i.e., the largest value of $K_k/K$

- ***To classify a new point, we identify the $K$ nearest points from the training data set and then assign the new point to the class having the greatest number of representatives amongst the $K$ nearest neighbor set.***

# The KNN Classification Algorithm

**Begin**

load the training data set $\{\boldsymbol{x}_i, y_i\}$, and the test data point $\boldsymbol{x}$

Choose the value of $K$, set $\mathcal{D} = \{\}$

**For** $i = 1$ to $N$

calculate $d_i = \|\boldsymbol{x} - \boldsymbol{x}_i\|$

append $\{y_i, d_i\}$ to $\mathcal{D}$

**End**

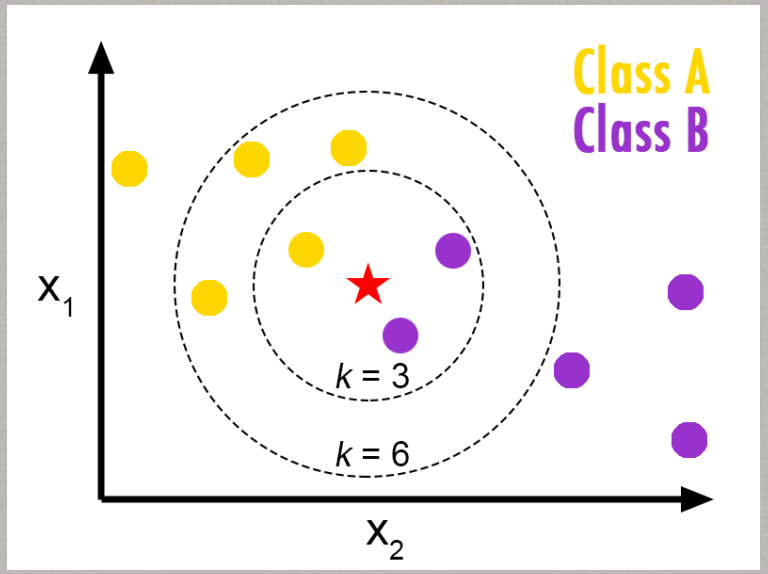
sort $\mathcal{D}$ in the ascend order with respect to $d$ value

pick the first $K$ entries $\{y_n, d_n\}, n = 1, \dots, K$ from $\mathcal{D}$

$y = mode(\{y_n, d_n\}, n = 1, \dots, K)$

**Return** $y$

**End**



# Metrics used in the KNN method

- ***Distance functions*** are used in KNN method as measure of similarity of instances

- Commonly used distance functions in machine learning include the following:

Let $\boldsymbol{x} = [x_1 \quad \cdots \quad x_m]^T$ *and* $\boldsymbol{y} = [y_1 \quad \cdots \quad y_m]^T$ be two vectors, then the distances between $\boldsymbol{x}$ *and* $\boldsymbol{y}$ are:

- *Euclidean* distance

$$d(\boldsymbol{x}, \boldsymbol{y}) = \left( \sum_{i=1}^{m} (x_i - y_i)^2 \right)^{\frac{1}{2}}$$

$L_2$ norm

- *Minkowsky* distance

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

$L_p$ norm

- *Manhantan* distance

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

L1 norm

- *Chebyshev* distance

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

L∞ norm

- *Hamming* distance:

$$d_i = \begin{cases} 1, & if\ x_i \neq y_i \\ 0, & otherwise \end{cases};\ d(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=1}^{m} d_i$$

Let $\boldsymbol{x} = [x_1 \quad \cdots \quad x_m]^T$ *and* $\boldsymbol{y} = [y_1 \quad \cdots \quad y_m]^T$ be two vectors, we have,

- Cosine similarity measure:

$$sim(\boldsymbol{x}, \boldsymbol{y}) = \frac{\langle \boldsymbol{x}, \boldsymbol{y} \rangle}{\|\boldsymbol{x}\|\|\boldsymbol{y}\|} = \frac{\sum_{i=1}^{m} x_i y_i}{\sqrt{\sum_{i=1}^{m} x_i^2} \sqrt{\sum_{i=1}^{m} y_i^2}}$$

- Correlation:

$$corr(\boldsymbol{x}, \boldsymbol{y}) = \frac{\sum_{i=1}^{m} (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2} \sqrt{\sum (y_i - \bar{y})^2}} = \frac{\langle \boldsymbol{x} - \bar{\boldsymbol{x}}, \boldsymbol{y} - \bar{\boldsymbol{y}} \rangle}{\|\boldsymbol{x} - \bar{\boldsymbol{x}}\|\|\boldsymbol{y} - \bar{\boldsymbol{y}}\|} = sim(\boldsymbol{x} - \bar{\boldsymbol{x}}, \boldsymbol{y} - \bar{\boldsymbol{y}})$$

# Questions about the KNN method

**Q1:** How do we train a KNN algorithm?

**Q2:** Does a KNN algorithm need a training set/test set? Yes, validation set → K, distance

**Q3:** Can the KNN method be applied to the regression problem? Yes, average

**Q4:** Is KNN a good candidate for online (real-time) classification? Not

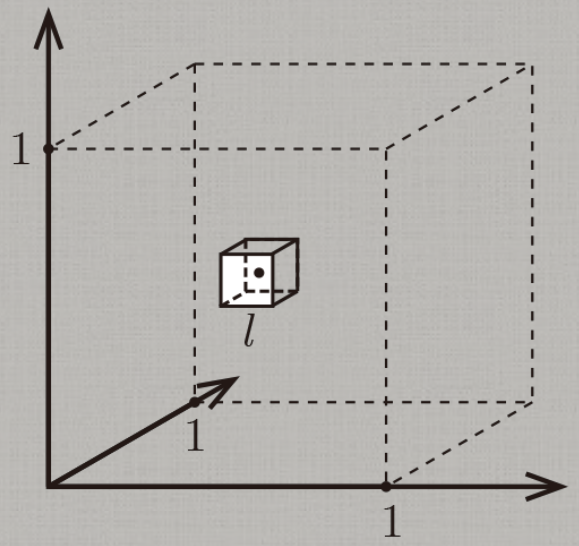
# The curse of dimensionality 维度诅咒

- The loss of locality in ***higher dimensions***
  - Consider $N$ data points uniformly distributed in a $d$ dimensional unit hypercube.
  - Let's find out a hypercube of edge $l$ that contains $K$ nearest neighbors of a test point. That is, we want to find a cube of edge $l$ to contain a fraction $r = \frac{K}{N}$ of the entire data set.
  - Since the data points are uniformly distributed, this means the volume of the cube is fraction $r$ of the unit cube, i.e.,

$$l^d = r \Rightarrow l = r^{\frac{1}{d}}$$

  - When $d \to \infty$, we have $l \to 1$. i.e., the hypercube is close to the entire unit hypercube where the data is uniformly distributed!!! This is true even for very small $r$ value.

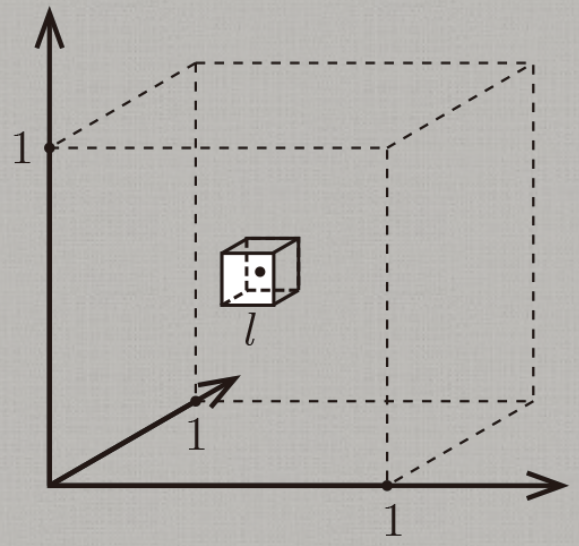
$$0 < l < 1$$



For example, let $N = 1000, K = 10$, we have, $r = 0.01$

$$l^d = r \Rightarrow l = r^{\frac{1}{d}}$$

| $d$ | $l$ |
|---|---|
| 1 | 0.01 |
| 2 | 0.1 |
| 3 | 0.21 |
| 10 | 0.63 |
| 100 | 0.96 |
| 1000 | 0.995 |




To find the 10 – nearest neighbor, we need almost the entire input space in high dimensions!!!

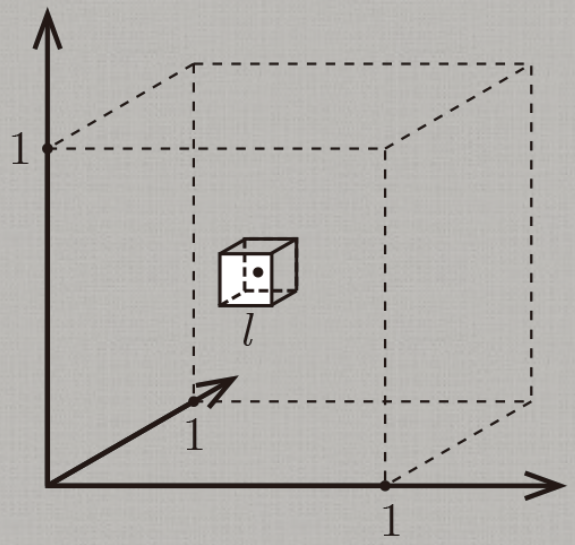
- To find the 10 – nearest neighbor, we need almost the entire input space in high dimensions!!!
- This breaks down the KNN assumptions because in high dimensions, the points in $K$ nearest neighbor are not particularly closer (similar) than any other data points in the training set.
- Can we increase the number of training data points until the $K$ nearest neighbors are truly closer to the test point?
- How many data points do we need such that $l$, the size of the neighborhood become truly small?

For example,

let $l = 0.1$. then, $r = \frac{K}{N} = l^d \Rightarrow N = \frac{K}{l^d} = K \times 10^d$

When $d = 10$, $N$ becomes a huge number!!!

Dimension reduction (PCA) may provide a solution to alleviate the problem!



Quiz

# The Naïve Bayes Classifier – A generative approach

- The training data set is given as $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{N}$, where, an instance $\boldsymbol{x}_i$ belongs to class $\mathcal{C}_k$ if $y_i = k, k = 1, \ldots, K$.
- Given a new instance, $\boldsymbol{x} = [x_1 \quad x_2 \quad \cdots \quad x_d]^T$, we wish to predict the class of this instance.
- The ***Bayesian approach*** to classify the new instance is to assign the most probable target value, $y_{MAP}$, given the attribute values $x_1, \ldots, x_d$. That is,

屬性值

$$y_{MAP} = \underset{k}{argmax}\, P(\mathcal{C}_k | x_1, \ldots, x_d)$$

maximum a posteriori (MAP)

最大后验

$\underset{x}{argmax}\, f(x)$：使 $f(x)$ 最大的 $x$ 值

- Using ***Bayes' theorem***, this probability can be written as:

求的是最大K值

$$\underset{k}{argmax}\, P(C_k|x_1,\cdots,x_d) = y_{MAP} = \underset{k}{argmax}\frac{P(x_1,\ldots,x_d|C_k)P(C_k)}{P(x_1,\ldots,x_d)}$$

不依赖于类别 $C_k$
对于不同 $C_k$，$P(X)$ 相同
数据集大时会很小

$$= \underset{k}{argmax}\, P(x_1,\ldots,x_d|C_k)P(C_k)$$

- ***Assume the attributes are conditionally independent given the target value***, we have

$$P(x_1,\ldots,x_d|C_k) = P(x_1|C_k)P(x_2|C_k)\cdots P(x_d|C_k) = \prod_{j=1}^{d} P(x_j|C_k)$$

- Then we have,

$$y_{MAP} = \underset{k}{argmax}\, P(C_k)\prod_{j=1}^{d} P(x_j|C_k)$$

需要计算每个类的该值，
选最大值，就是该类

- This is why the method is called "***naïve***" Baye's method

先根据已知数据计算各 $P(C_k)$

再计算各 $P(x_1, \cdots, x_d | C_k) = P(x_1|C_k) \cdots P(x_d|C_k)$

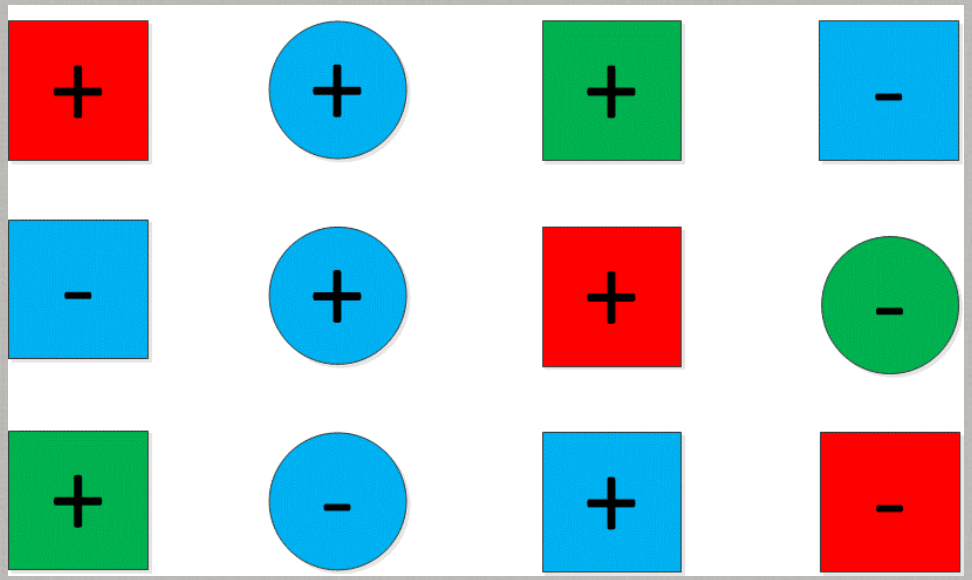
## The Naïve Bayes Classifier – A Toy Example

根据以上两数值可得各 $P(C_k | x_1, \cdots, x_d) = \frac{P(x_1, \cdots, x_d | C_k) P(C_k)}{P(x_1, \cdots, x_d)}$

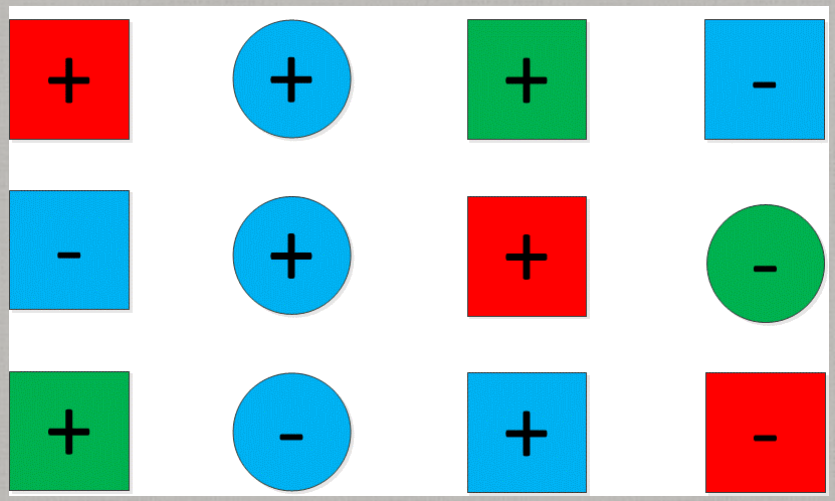
对比大小选出最大的 $P(C_k | x_1, \cdots, x_d)$

$P(x_1, \cdots, x_d)$ 相同

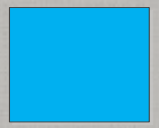
- The training data set is given as:




- $y_i \in \{+, -\}, i = 1,2, \dots, 12$ are the class labels.
- $\boldsymbol{x}_i = [x_{i1} \quad x_{i2}]^T, x_{i1} \in \{blue, green, red, yellow\}, x_{i2} \in \{square, circle\}, i = 1, \dots, 12$ are the feature vectors.




- Given the following new input, let's use ***naïve Bayes method*** to predict its class:



- The feature vector is $\boldsymbol{x} = [x_1, x_2]^T = [blue, square]^T$

Step 1: $P(+) = \frac{7}{12}$ , $P(-) = \frac{5}{12}$

Step 2: $P(\vec{x}|+) = P(x_1 = \text{blue}, x_2 = \text{square}|+)$

$= P(x_1 = \text{blue}|+) \cdot P(x_2 = \text{square}|+)$

小心数错 $= \frac{3}{7} \times \frac{5}{7} = 0.31$

$P(\vec{x}|-) = P(x_1 = \text{blue}, x_2 = \text{square}|-)$

$= P(x_1 = \text{blue}|-) \cdot P(x_2 = \text{square}|+)$

$= \frac{3}{5} \times \frac{3}{5} = 0.36$

Step 3: $P(+|\vec{x}) \overset{\propto}{=} P(\vec{x}|+) P(+) = 0.31 \times \frac{7}{12} = 0.18$

$P(-|\vec{x}) = P(\vec{x}|-) P(-) = 0.36 \times \frac{5}{12} = 0.15$

Step 4: $P(+|\vec{x}) > P(-|\vec{x})$

Thus, blue square is classified into class +

- Under the assumption that the training samples are $i.i.d$, we have the class priors as:

$$P(+) = \frac{7}{12} = 0.58; \quad P(-) = \frac{5}{12} = 0.42$$

$P(C_k)$

- ***Assuming features are conditionally independent for given class***, the class conditional probabilities can be calculated as:

$$P(\boldsymbol{x}|+) = P(x_1 = blue, x_2 = square|+) = P(blue|+)P(square|+) = \frac{3}{7} \times \frac{5}{7} = 0.31$$

$$P(\boldsymbol{x}|-) = P(x_1 = blue, x_2 = square|-) = P(blue|-)P(square|-) = \frac{3}{5} \times \frac{3}{5} = 0.36$$

$$P(+|x) = \frac{P(x|+)\,P(+)}{P(x)}$$

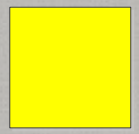
- Then the posterior can be calculated as:

$$P(+|\boldsymbol{x}) = P(\boldsymbol{x}|+)P(+) = 0.58 \times 0.31 = 0.18$$

$$P(-|\boldsymbol{x}) = P(\boldsymbol{x}|-)P(-) = 0.42 \times 0.36 = 0.15$$

- Since $P(+|\boldsymbol{x}) = 0.18 > P(-|\boldsymbol{x}) = 0.15$, the sample point $\boldsymbol{x}$ =  is classified as $+$.

- Suppose a new sample as the following is given:



- The feature vector is $\boldsymbol{x} = [x_1, x_2]^T = [yellow, square]^T$, the color attribute value "yellow" is not present in the training data set. This causes the class conditional probability to become:

$$P(\boldsymbol{x}|+) = P(x_1 = yellow, x_2 = square|+) = P(yellow|+)P(square|+) = 0 \times \frac{5}{7} = 0$$

$$P(\boldsymbol{x}|-) = P(x_1 = yellow, x_2 = square|-) = P(yellow|-)P(square|-) = 0 \times \frac{3}{5} = 0$$

And the posteriors will hence be all calculated as 0s.

- This problem can be solved by using the ***Laplace smoothing*** (***additive smoothing***) technique to calculate the class conditional probability:

$$P(x_i|C_k) = \frac{N_{x_i,C_k} + 1}{N_{C_k} + d}$$

特征 $x_i$ 在类别 $C_k$ 训练集中出现的次数

where, $N_{x_i,C_k}$ is the number of time feature $x_i$ appears in training data set from class $C_k$;

$C_k$ 类训练集中所有特征出现的总数

$N_{C_k}$ is the total count of all features appears in training data set from class $C_k$; $d$ is the dimensionality of the feature vector

特征向量 $x$ 的维度

- Then, using this new technique, the class conditional probability of the new sample $\boldsymbol{x}$ can be calculated as:

$$P(\boldsymbol{x}|+) = P(x_1 = yellow, x_2 = square|+) = P(yellow|+)P(square|+)$$

$$= \frac{1}{7+2} \times \frac{5}{7} = 0.08$$

$$P(\boldsymbol{x}|-) = P(x_1 = yellow, x_2 = square|-) = P(yellow|-)P(square|-)$$

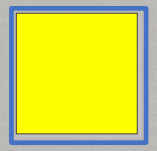
$$= \frac{1}{5+2} \times \frac{3}{5} = 0.09$$

- *Note that the Laplace smoothing formula is used to calculate the likelihood of the features that do not present in the training data set.*

- The posterior probability of the classes can be calculated as:

$$P(+|\boldsymbol{x}) = P(\boldsymbol{x}|+)P(+) = 0.08 \times 0.58 = 0.0464$$

$$P(-|\boldsymbol{x}) = P(\boldsymbol{x}|-)P(-) = 0.09 \times 0.42 = 0.0378$$

Therefore, the new sample  should be classified as $+$

# The Naïve Bayes Classifier – Text Documents Classification

- ***Inputs:***
  - a document $\boldsymbol{x}$
  - A fixed set of classes $\{\mathcal{C}_1, \dots, \mathcal{C}_K\}$ e.g., {"spam", "ham"}, {"computer science", "biology", "statistics", "economics", "politics"}
  - A training set of $N$ hand-labeled documents $\{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \dots, (\boldsymbol{x}_N, y_N)\}$ (training corpus)
- ***Output:***
  - The class of the document $\boldsymbol{x}$

Example: Email

# Preprocessing of text documents

- ***Tokenization**: breaks down a text corpus into individual words and removes punctuations*

语料库

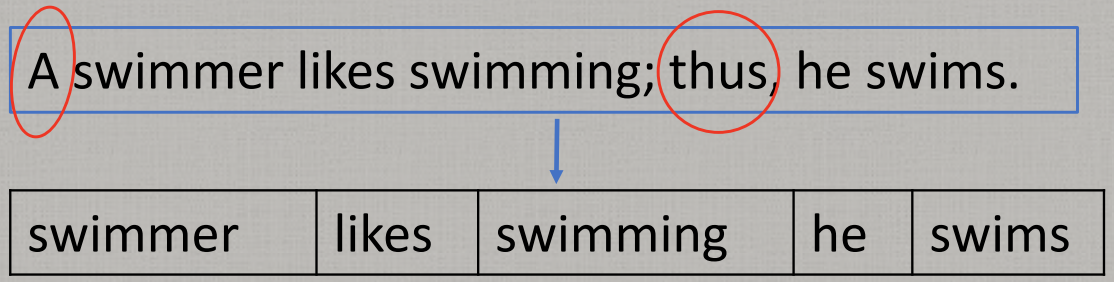
A swimmer likes swimming; thus, he swims.



| a | swimmer | likes | swimming | thus | he | swims |
|---|---|---|---|---|---|---|

# Preprocessing of text documents

- ***Remove stop words***: stop words are particularly common in a text corpus and thus considered as rather uninformative (e.g., words such as ***so, and, or, the***, etc.) a stop list can be created and used for stop words removal.



# The Bag of Words Model

- ***Creation of the vocabulary***: after preprocessing, we can create the collection of all different words that occur in the training data set and each word is associated with a count of how it occurs. ***This is a set of non-redundant items where the order does not matter (bag of words)***.

多余的

Let $D_1$ $and$ $D_2$ be two documents in a training dataset (corpus):

- $D_1$: "Each state has its own laws."
- $D_2$: "Every country has its own culture."

Based on these two documents, after proper preprocessing, the vocabulary can be created as:

$$V = \{each: 1, state: 1, has: 2, its: 2, own: 2, laws: 1, every: 1, country: 1, culture: 1\}$$

- ***Vectorization:*** The vocabulary can then be used to construct the *d*-dimensional feature vector for the individual documents. $d = |V|$ is the number of different words in the vocabulary.

- The following are the ***bag of words*** representation of two sample documents $D_1$ $and$ $D_2$: (word position information does not matter)

2 documents to 2 vectors

| $V$ | each | state | has | its | own | laws | every | country | culture |
|---|---|---|---|---|---|---|---|---|---|
| $\boldsymbol{x}_{D_1}$ | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| $\boldsymbol{x}_{D_2}$ | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 |
| $\Sigma$ | 1 | 1 | 2 | 2 | 2 | 1 | 1 | 1 | 1 |

- After transforming documents into numbers, popular supervised learning methods can be applied to do document classification.

# Text Documents Classification using The Naïve Bayes Classifier

- Given a test document $\boldsymbol{x} = [x_1, x_2, \dots, x_d]^T$
- ***Prior probability estimate:***

$$P(\mathcal{C}_k) = \frac{\#\ of\ documents\ with\ \mathcal{C}_k}{Total\ \#\ of\ documents\ in\ the\ training\ data\ set}, \qquad k = 1,2, \dots, K$$

- ***Class conditional probability estimate: (Laplace smoothing if necessary)***

$$P(x_i|\mathcal{C}_k) = \frac{N(x_i, \mathcal{C}_k) + 1}{N(\mathcal{C}_k) + |V|}$$

Where, $N(x_i, \mathcal{C}_k)$ is the total number of occurrences of term $x_i$ in all documents in the training data set with class $\mathcal{C}_k$; $N(\mathcal{C}_k)$ is the number of occurrences of all the terms in all documents in the training data set with class $\mathcal{C}_k$.

- ***Class conditional probability estimate:***

$$P(x_i|\mathcal{C}_k) = \frac{N(x_i, \mathcal{C}_k) + 1}{N(\mathcal{C}_k) + |V|}$$

Where, $N(x_i, \mathcal{C}_k)$ is the total number of occurrences of term $x_i$ in all documents in the training data set with class $\mathcal{C}_k$; $N(\mathcal{C}_k)$ is the number of occurrences of all the terms in all documents in the training data set with class $\mathcal{C}_k$.

- Then, assuming features are conditionally independent given class, we have, ("naïve")

$$P(\boldsymbol{x}|\mathcal{C}_k) = P(x_1, x_2, \dots, x_d|\mathcal{C}_k) = \prod_{i=1}^{d} P(x_i|\mathcal{C}_k)\,, k = 1, \dots, K$$

***The chance of occurrence of value $x_i$ is not affected by the values of other attributes.***

- ***Calculation of the posterior probability:***

$$P(\mathcal{C}_k|x) \propto P(\boldsymbol{x}|\mathcal{C}_k)P(\mathcal{C}_k)$$