Aprendizado por Memorização: IBL & K-NN

- Instance-Based Learning (IBL) é um paradigma no qual os algoritmos tipicamente armazenam alguns ou todos os exemplos de treinamento durante o aprendizado
- Para classificar um novo exemplo, estes sistemas utilizam uma função de distância ou similaridade para determinar o quão próximo o novo exemplo encontra-se de um exemplo já armazenado e utiliza este(s) exemplo(s) mais próximo(s) para determinar a classe do novo exemplo
- Veremos nesta aula o funcionamento de IBL, K-NN e algumas métricas de distância

José Augusto Baranauskas
Departamento de Física e Matemática – FFCLRP-USP
augusto@usp.br
http://dfm.ffclrp.usp.br/~augusto

Conteúdo

Instance-Based Learning

- Instance-Based Learning (IBL): aprendizado por memorização de exemplos
- Memorização: aprende conceitos apenas armazenando exemplos típicos do conceito
- Não constrói representações abstratas
- Geralmente, requer grande capacidade de memória
- Exemplos: IBL1, IBL2 e IBL3

Exemplo IBL

- Problema: movimento de um braço de robô
  - Modelo difícil de ser analisado analiticamente
    - Equações cinemáticas
      - Relacionam ângulos de junções e manipulador de posições
    - Equações dinâmicas
      - Relacionam torque de motor a ângulos das juntas
  - Difícil obter bons resultados modelando braços robóticos ou humanos
    - Muitos fatores e medidas

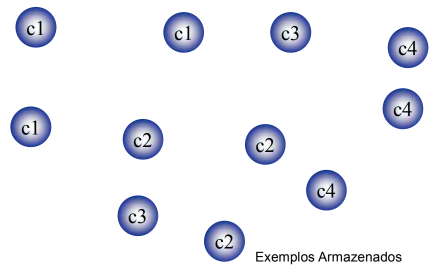
Exemplo IBL

- Solução
  - Movimente o braço do robô
  - Armazene os parâmetros bem como a trajetória
    - Tabela: torques, posições, velocidades, velocidade ao quadrado, acelerações
  - Para seguir um novo caminho:
    - Particione o caminho em segmentos
    - Encontre os segmentos mais próximos na tabela
    - Utilize os torques (interpole quando necessário)

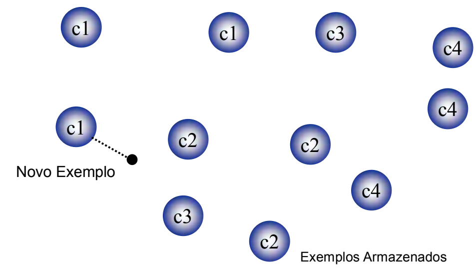
IBL1

- Aprende um conceito simplesmente armazenando todos os exemplos
- Um novo exemplo é classificado
  - calculando a distância Euclidiana para determinar o vizinho mais próximo
  - classe do novo exemplo é dada pela classe do vizinho mais próximo
- ⮝ Tolerante a ruídos
- ⮟ Usa muita memória

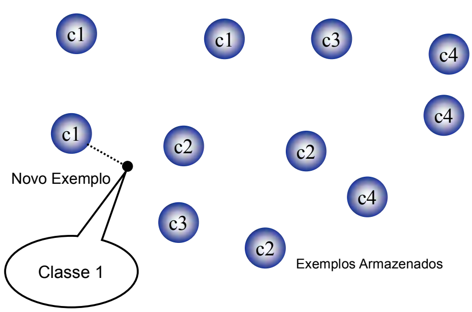
Algoritmo IBL1



Algoritmo IBL1



Algoritmo IBL1



Algoritmo IBL1

- ❑ Aprendizado: IBL1-Learn(T)
  - Tabela ← armazene todos os n exemplos de T
  - Retorne Tabela
- ❑ Classificação: IBL-Classify(Tabela,E)
  - Calcule a distância entre o exemplo E e todos os demais exemplos da Tabela para determinar o vizinho mais próximo P
  - if P não existe then
  Retorne NULL
else
  Retorne classe(P)
endif

IBL2

- ❑ Idem ao IBL1 exceto:
- ❑ Objeto novo:
  - classe correta: ignora (não armazena)
  - classe errada: armazena

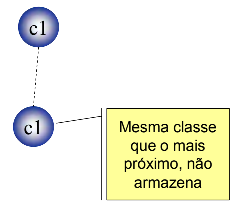
- ⋎ Menos tolerante a ruídos que IBL1
- ⅄ Usa menos memória que IBL1

Algoritmo IBL2

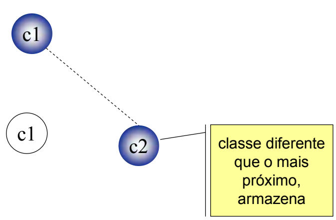
- ❑ Aprendizado: IBL2-Learn(T)
  - Tabela ← Ø
  - for i ← 1,2,…,n do
    - ❖ Assuma exemplo no formato $E_i = (x_i, y_i)$, ou seja, y_i = classe(E_i)
    - ❖ if IBL-Classify(Tabela,x_i) $\neq y_i$ then
  Tabela ← Tabela ∪ {E_i}
    - ❖ endif
  - endfor
  - Retorne Tabela
- ❑ Classificação: IBL-Classify(Tabela,E)
  - Calcule a distância entre o exemplo E e todos os demais exemplos da Tabela para determinar o vizinho mais próximo P
  - if P não existe then
  Retorne NULL
else
  Retorne classe(P)
endif

Algoritmo IBL2

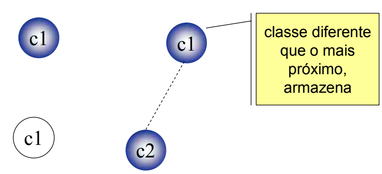
Algoritmo IBL2



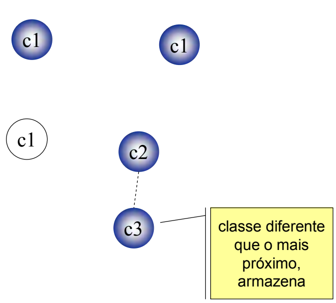
Algoritmo IBL2



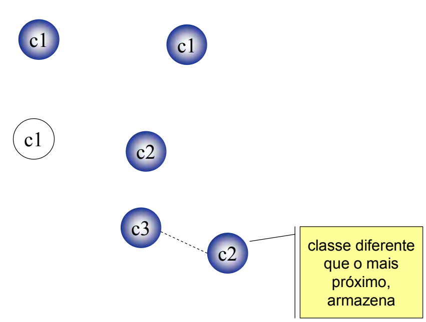
Algoritmo IBL2



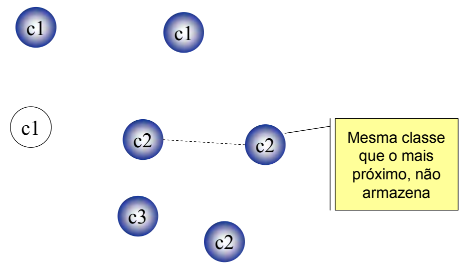
Algoritmo IBL2



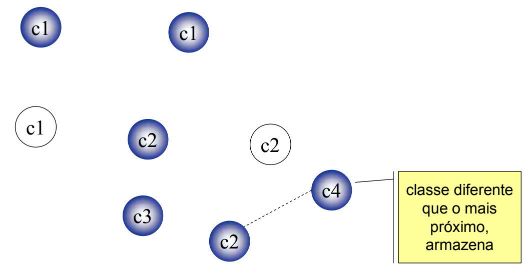
Algoritmo IBL2



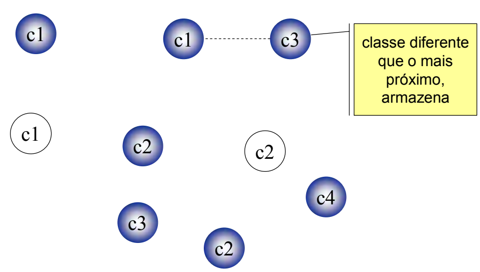
Algoritmo IBL2



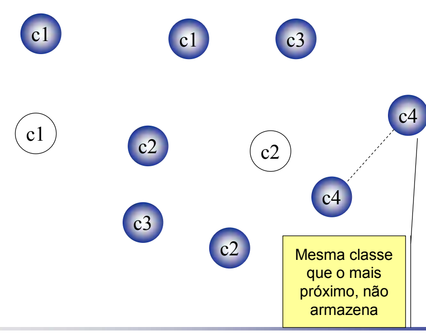
Algoritmo IBL2



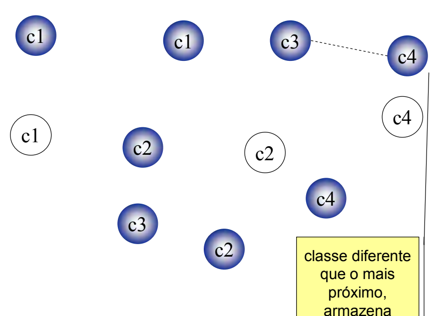
Algoritmo IBL2



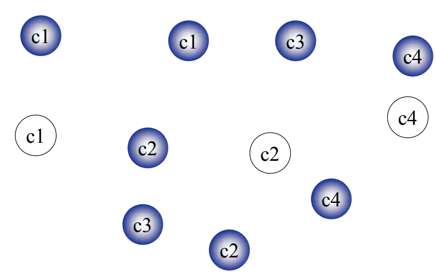
Algoritmo IBL2



Algoritmo IBL2



Algoritmo IBL2



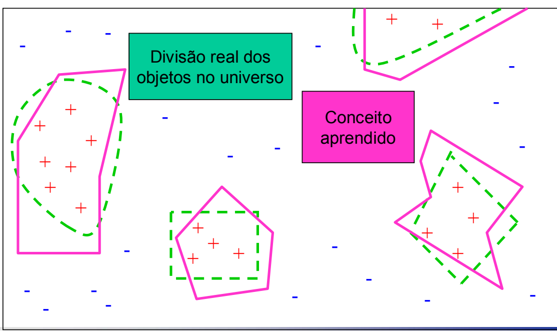
IBL3

- Idem ao IBL2 exceto:
- Mantém um registro do número de classificações corretas e incorretas para cada exemplo armazenado
- Este registro mantém o desempenho de classificação daquele exemplo
- IBL3 avalia cada exemplo, utilizando um teste de significância para determinar quais exemplos são bons classificadores e quais são ruídos
- Exemplos com ruído são descartados
- Tolerante a ruídos
- Usa menos memória que IBL2

IBL1 x IBL2 x IBL3

- IBL1 fortemente relacionado com K-vizinhos mais próximos (K-NN, K-Nearest Neighbors), K=1
- IBL2 e IBL3 usam métodos de *esquecimento* de exemplos (exemplos que não melhoram a precisão do classificador)

Fronteiras IBL



Algoritmo K-NN

- Normalmente utiliza distância Euclidiana entre os exemplos
- Encontre os **K** vizinhos mais próximos do novo exemplo a ser classificado
- Retorne a classe associada à maioria dos vizinhos, ou seja, se a maioria dos exemplos encontrados entre os **K** vizinhos possui classe C_i então atribua ao novo exemplo a classe C_i
  - Usualmente valores ímpares são utilizados para evitar empate, tipicamente K = 1, 3, 5 ou 7
  - Quanto maior a quantidade de ruído no conjunto de exemplos, maior deve ser o valor de K

Algoritmo K-NN

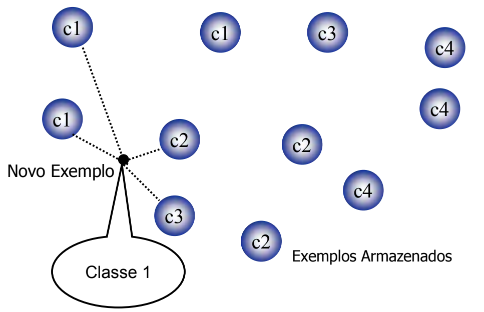
- Uma variação do algoritmo é:
  - Seja um **t** (pré-definido ou definido pelo usuário) tal que (0 < **t** <= **K**)
  - Encontre os **K** vizinhos mais próximos do novo exemplo a ser classificado
  - Se pelo menos **t** (0 < **t** <= **K**) dos exemplos encontrados entre os **K** vizinhos possui classe C_i então atribua ao novo exemplo a classe C_i, caso contrário o exemplo não é classificado (ou seja, o algoritmo não retorna a classe associada)

Algoritmo K-NN

- No caso de regressão, normalmente é efetuada uma média entre os **K** vizinhos mais próximos do novo exemplo
- Denotando o novo exemplo **E**=(x_e,y_e) e os **K** vizinhos mais próximos de **E** por (x_i^{KNN}, y_i^{KNN}), i=1,2,..., **K** então o valor de $\hat{y}_e$ é calculado como

$$h(x_e) = \hat{y}_e = \frac{1}{K}\sum_{i=1}^{K} y_i^{KNN}$$

Algoritmo K-NN



Exemplo: Classificação

- Assuma os seguintes exemplos de treinamento
  - $x_1 = \langle 1,2,7,8,+\rangle$
  - $x_2 = \langle 1,3,7,6,-\rangle$
- Classifique $x_3 = \langle 1,2,7,6\rangle$
  - Calcular distância entre x_3 e cada um dos exemplos de treinamento
    - $dist(x_1, x_3) = 2$
    - $dist(x_2, x_3) = 1$
  - Classifique (rotule) x_3 baseado nos K vizinhos mais próximos, onde K=1
    - x_3 é classificado como – uma vez que é mais próximo a x_2

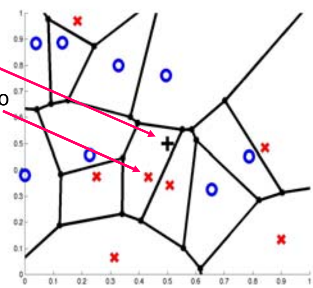
Exemplo: Regressão

- Assuma os seguintes exemplos de treinamento
  - $(x_1,1)$
  - $(x_2,1)$
  - $(x_3,0)$
- Desejamos classificar x_4 usando K=2 vizinhos mais próximos, sabendo que
  - $dist(x_1,x_4) = 5$
  - $dist(x_2,x_4) = 2$
  - $dist(x_3,x_4) = 4$
- Portanto x_4 está mais próximo a x_2 e x_3
- Calculando a média, obtém-se (1+0)/2 = 0.5

Diagrama de Voronoi

Novo Exemplo x

Vizinho mais Próximo de x



Exemplo: problema de 2 classes (x,o) e 2 atributos representados nos eixos x e y

Diagrama de Voronoi, K=3

Novo Exemplo

3 vizinhos mais próximos

2x,1o

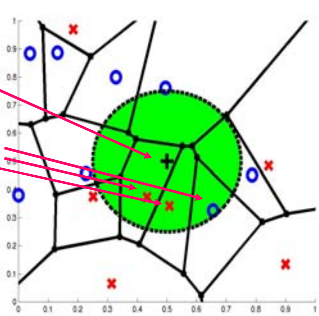
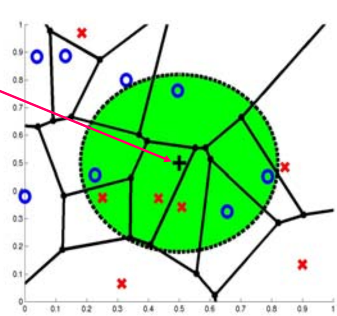


Diagrama de Voronoi, K=7

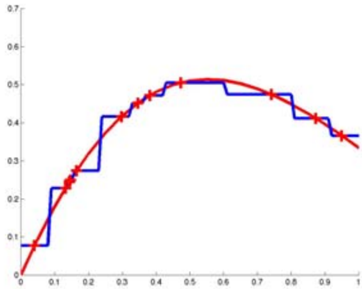
Novo Exemplo

7 vizinhos mais próximos

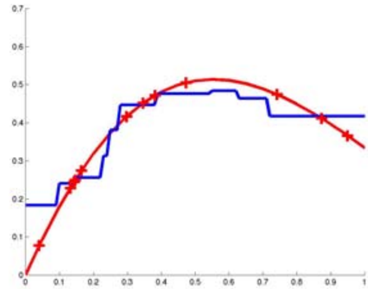
3x,4o



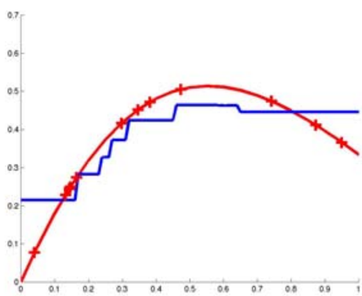
KNN: Regressão, K=1



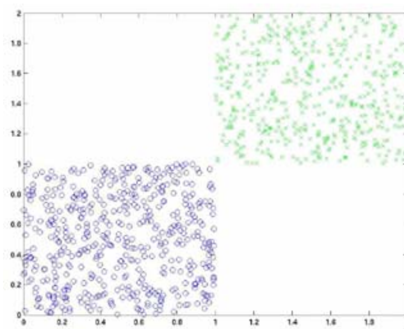
KNN: Regressão, K=3



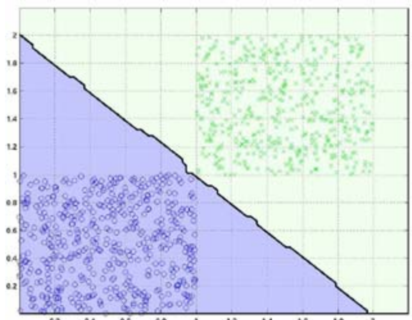
KNN: Regressão, K=5



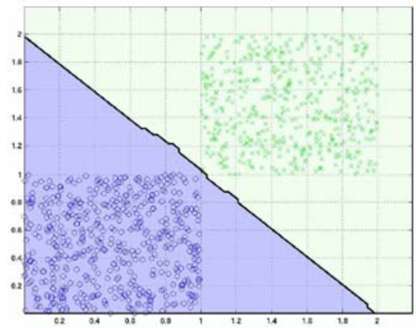
Fronteiras



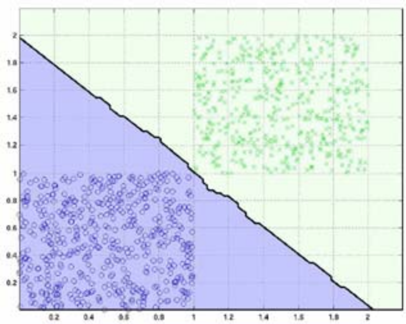
Fronteiras, K=1



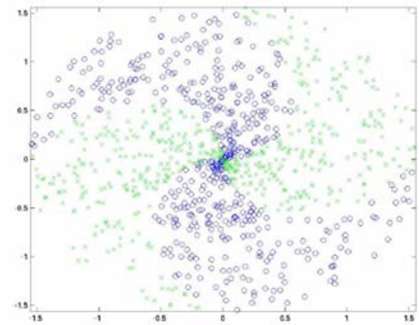
Fronteiras, K=5



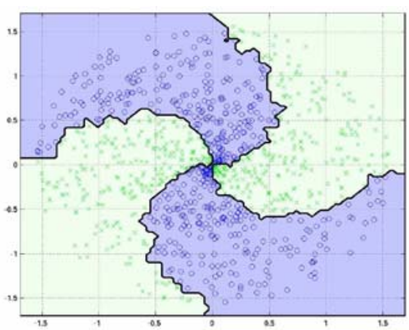
Fronteiras, K=10



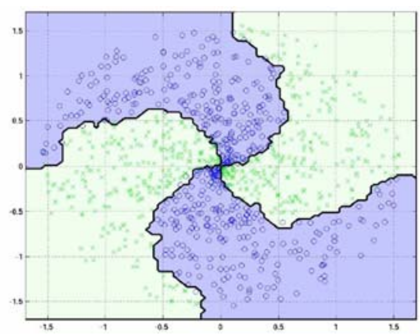
Fronteiras



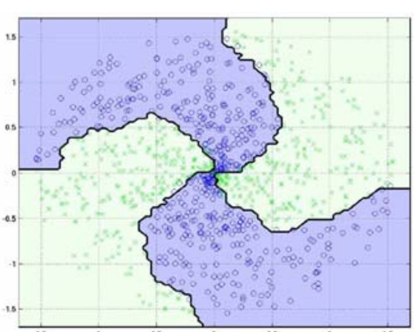
Fronteiras, K=1



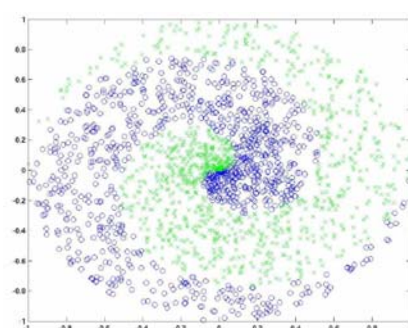
Fronteiras, K=5



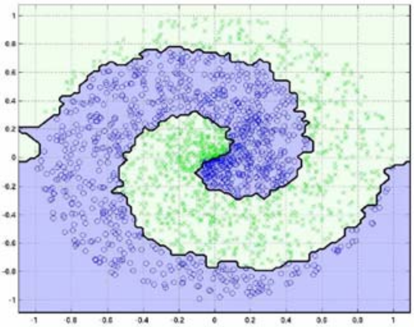
Fronteiras, K=10



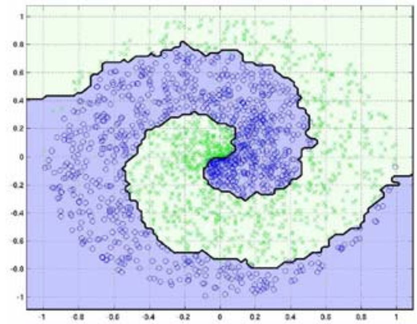
Fronteiras



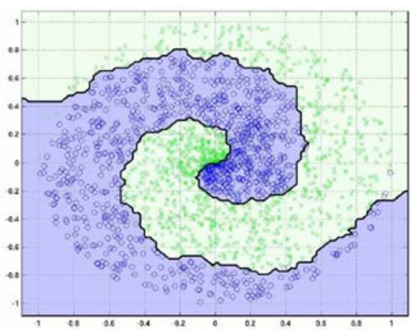
Fronteiras, K=1



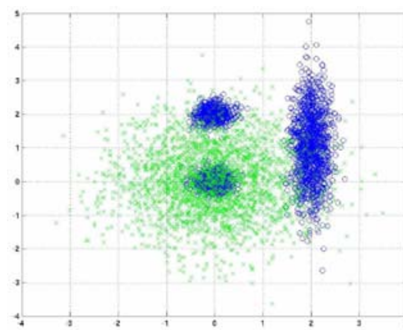
Fronteiras, K=5



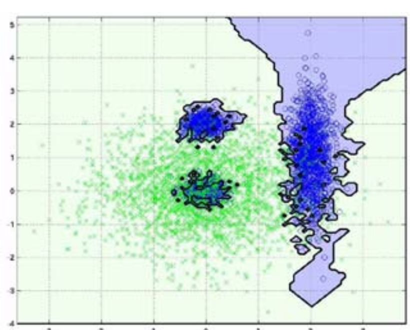
Fronteiras, K=10



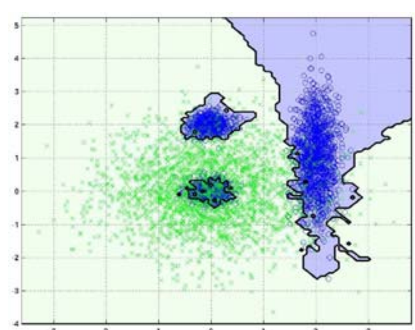
Fronteiras



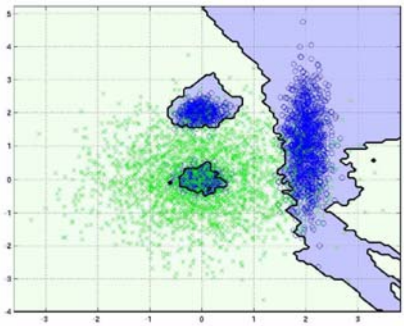
Fronteiras, K=1



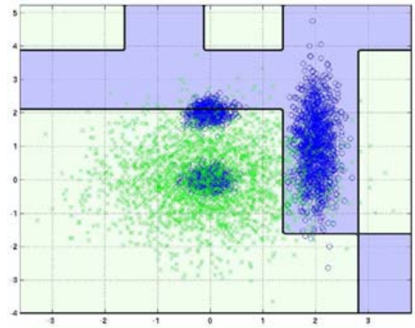
Fronteiras, K=5



Fronteiras, K=10



Fronteiras (Árvore de Decisão)



Análise de Complexidade

- Custo de Aprendizado O(n)
  - Armazenar exemplos
- Custo de Classificação
  - Encontrar o vizinho mais próximo: O(n)
    - Computar distância entre o novo exemplo e todos os demais exemplos
    - Comparar distâncias
  - Problemático para grande conjunto de exemplos
- Alternativa:
  - Usar uma árvore binária de busca para reduzir para O(log n)

Calculando a Distância

- A distância é o método mais natural para dados numéricos
- Valores pequenos indicam maior similaridade
- Métricas de Distância
  - Euclidiana
  - Manhattan
  - Etc.
- Não generaliza muito bem para dados não numéricos
  - Qual a distância entre “masculino” e “feminino”?

Normalização

- Considerando a distância Euclidiana, mais utilizada nas aplicações, um problema ocorre quando um dos atributos assume valores em um intervalo relativamente grande, podendo sobrepujar os demais atributos
- Por exemplo, se uma aplicação tem apenas dois atributos A e B e A varia entre 1 e 1000 e B entre 1 e 10, então a influência de B na função de distância será sobrepujada pela influência de A
- Portanto, as distâncias são freqüentemente **normalizadas** dividindo a distância de cada atributo pelo intervalo de variação (i.e. diferença entre valores máximo e mínimo) daquele atributo
- Assim, a distância para cada atributo é **normalizada** para o intervalo [0,1]

Normalização

- De forma a evitar ruídos, é também comum:
  - dividir pelo desvio-padrão ao invés do intervalo ou
  - “cortar” o intervalo por meio da remoção de uma pequena porcentagem (e.g. 5%) dos maiores e menores valores daquele atributo e somente então definir o intervalo com os dados remanescentes
    - Também é possível mapear qualquer valor fora do intervalo para os valores mínimo ou máximo para evitar valores normalizados fora do intervalo [0,1]
- Conhecimento do domínio pode freqüentemente ser utilizado para decidir qual método é mais apropriado

Métricas

$$\mathbf{x_i} = (x_{i,1}, x_{i,2}, x_{i,3}, \cdots, x_{i,m})$$
$$\mathbf{x_j} = (x_{j,1}, x_{j,2}, x_{j,3}, \cdots, x_{j,m})$$

- Minkowski (L_p): escolha de p depende da ênfase que se deseja dar a grandes diferenças entre dimensões

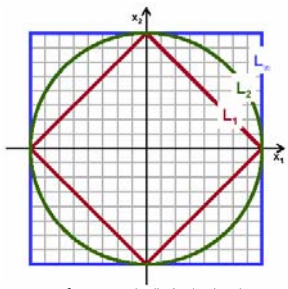
$$dist_p(\mathbf{x_i}, \mathbf{x_j}) = \left[\sum_{r=1}^{m} |x_{i,r} - x_{j,r}|^p \right]^{1/p}$$

- Manhattan/City-Block (L_1): se atributos binários, é conhecida como distância Hamming

$$dist_M(\mathbf{x_i}, \mathbf{x_j}) = \sum_{r=1}^{m} |x_{i,r} - x_{j,r}|$$

- Euclidiana (L_2)

$$dist_2(\mathbf{x_i}, \mathbf{x_j}) = \left[\sum_{r=1}^{m} (x_{i,r} - x_{j,r})^2 \right]^{1/2}$$



Contornos de distâncias iguais

Métricas

$$\mathbf{x_i} = (x_{i,1}, x_{i,2}, x_{i,3}, \cdots, x_{i,m})$$
$$\mathbf{x_j} = (x_{j,1}, x_{j,2}, x_{j,3}, \cdots, x_{j,m})$$

- Camberra

$$dist_{\mathrm{Ca}}(\mathbf{x_i}, \mathbf{x_j}) = \sum_{r=1}^{m} \frac{|x_{i,r} - x_{j,r}|}{|x_{i,r} + x_{j,r}|}$$

- Chebychev

$$dist_{\mathrm{Ch}}(\mathbf{x_i}, \mathbf{x_j}) = \max_{r=1}^{m} |x_{i,r} - x_{j,r}|$$

- Correlação

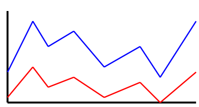
$$dist_{\mathrm{Co}}(\mathbf{x_i}, \mathbf{x_j}) = \frac{\sum_{r=1}^{m} (x_{i,r} - \bar{x}_i)(x_{j,r} - \bar{x}_j)}{\sqrt{\sum_{r=1}^{m} (x_{i,r} - \bar{x}_i)^2 \sum_{r=1}^{m} (x_{j,r} - \bar{x}_j)^2}}$$

$\bar{x}_i = \bar{x}_j$ média dos valores do atributo X_r

Métricas

$$\mathbf{x_i} = (x_{i,1}, x_{i,2}, x_{i,3}, \cdots, x_{i,m})$$
$$\mathbf{x_j} = (x_{j,1}, x_{j,2}, x_{j,3}, \cdots, x_{j,m})$$

- Correlação Pearson:
  - Remove efeitos de magnitude; intervalo [-1.0, 1.0]
  - -1.0 = inversamente correlacionado, 0.0 = sem correlação, 1.0 = perfeitamente correlacionado
- No exemplo, as linhas azul e vermelha têm alta correlação, mesmo que a distância entre as linhas seja significante



$$dist_{\mathrm{Pearson}}(\mathbf{x_i}, \mathbf{x_j}) = \frac{\sum_{r=1}^{m} x_{i,r} x_{j,r} - \left(\sum_{r=1}^{m} x_{i,r} \sum_{r=1}^{m} x_{j,r} \right) / m}{\sqrt{\left(\sum_{r=1}^{m} x_{i,r}^2 - (\sum_{k=1}^{m} x_{i,r})^2 / m \right) \left(\sum_{r=1}^{m} x_{j,r}^2 - (\sum_{r=1}^{m} x_{j,r})^2 / m \right)}}$$

Métricas

$$\mathbf{x_i} = (x_{i,1}, x_{i,2}, x_{i,3}, \cdots, x_{i,m})$$
$$\mathbf{x_j} = (x_{j,1}, x_{j,2}, x_{j,3}, \cdots, x_{j,m})$$

- O método mais simples para atributos categóricos é o seguinte

$$overlap(x_{i,r}, x_{j,r}) = \begin{cases} 1 & \text{se } x_{i,r} \text{ ou } x_{j,r} \text{ são desconhecidos} \\ 1 & \text{se } x_{i,r} \neq x_{j,r} \\ 0 & \text{se } x_{i,r} = x_{j,r} \end{cases}$$

$$dist_{\mathrm{Cat}}(\mathbf{x_i}, \mathbf{x_j}) = \sum_{r=1}^{m} overlap(x_{i,r}, x_{j,r})$$

Métrica Heterogênea

- Heterogeneous Euclidean-Overlap Metric: HEOM
- Utiliza normalização no intervalo [0,1]
- Uma forma de lidar com aplicações com atributos nominais e contínuos consiste em utilizar uma função de distância heterogênea que utiliza funções diferentes para tipos de atributos diferentes

$$dist_H(x_{i,r}, x_{j,r}) = \begin{cases} overlap(x_{i,r}, x_{j,r}) & \text{se atributo } X_r \text{ é nominal} \\ \dfrac{|x_{i,r} - x_{j,r}|}{máx(X_r) - mín(X_r)} & \text{se atributo } X_r \text{ é contínuo} \end{cases}$$

$$dist_{HEOM}(\mathbf{x_i}, \mathbf{x_j}) = \sqrt{\sum_{r=1}^{m} dist_H(x_{i,r}, x_{j,r})^2}$$

Métricas

- Value Difference Metric: VDM
- Atributos nominais
- Uma versão é dada a seguir
  - $N(X_r, x_{i,r})$: número de exemplos no conjunto T que tem o valor $x_{i,r}$ para o atributo X_r
  - $N(X_r, x_{i,r}, C_s)$: número de exemplos no conjunto T que tem o valor $x_{i,r}$ para o atributo X_r e classe igual a C_s
  - q: uma constante, usualmente 1 ou 2

$$dist_{VDM}(x_{i,r}, x_{j,r}) = \sum_{s=1}^{k} \left| \frac{N(X_r, x_{i,r}, C_s)}{N(X_r, x_{i,r})} - \frac{N(X_r, x_{j,r}, C_s)}{N(X_r, x_{j,r})} \right|^q$$

Métricas

- Usando VDM, dois valores são considerados próximos se eles possuem mais classificações similares (i.e. maior correlação com a classe), desconsiderando qualquer ordem que eles possam ter
- Por exemplo, se um atributo possui três valores *vermelho*, *verde* e *azul* e a aplicação consiste em identificar se um objeto é ou não uma maçã, *vermelho* e *verde* serão considerados mais próximos entre si do que *vermelho* e *azul*, uma vez que *vermelho* e *verde* têm correlação similar com a classe

Métricas

- Como vimos, a distância Euclidiana é inapropriada para atributos nominais e VDM é inapropriada para atributos contínuos
- Assim, nenhuma das métricas vistas é suficiente para uso em uma aplicação heterogênea, ou seja, uma com ambos atributos nominais e contínuos
- A *Heterogeneous Value Difference Metric* (HVDM) é uma função de distância heterogênea
- Para atributos contínuos, HVDM divide a diferença absoluta entre os atributos pelo desvio padrão do atributo (como 95% dos valores de uma distribuição normal estão entre 2 desvios-padrões, os valores são divididos por 4 desvios-padrões para mapear em um intervalo de tamanho 1)
- Para atributos discretos, HVDM pode utilizar VDM com q=1, q=2 (mas, na prática, a raiz quadrada não é tirada pois a função HVDM eleva os valores ao quadrado novamente) ou uma função utilizada em redes neurais *Radial Basis*

HVDM

$$dist_{HDVM}(\mathbf{x_i},\mathbf{x_j}) = \sqrt{\sum_{r=1}^{m} dist_{X_r}(x_{i,r},x_{j,r})^2}$$

$$dist_{X_r}(x_{i,r},x_{j,r}) = \begin{cases} 1 & \text{se } x_{i,r} \text{ ou } x_{j,r} \text{ são desconhecidos} \\ \text{norm-vdm}(x_{i,r},x_{j,r}) & \text{se } X_r \text{ é nominal} \\ \text{norm-diff}(x_{i,r},x_{j,r}) & \text{se } X_r \text{ é contínuo} \end{cases}$$

$$\text{norm-diff}(x_{i,r},x_{j,r}) = \frac{|x_{i,r} - x_{j,r}|}{4 \times \text{std-dev}(X_r)}$$

$$\text{norm-vdm1}(x_{i,r},x_{j,r}) = \sum_{s=1}^{k} \left| \frac{N(X_r,x_{i,r},C_s)}{N(X_r,x_{i,r})} - \frac{N(X_r,x_{j,r},C_s)}{N(X_r,x_{j,r})} \right|$$

$$\text{norm-vdm2}(x_{i,r},x_{j,r}) = \sqrt{\sum_{s=1}^{k} \left| \frac{N(X_r,x_{i,r},C_s)}{N(X_r,x_{i,r})} - \frac{N(X_r,x_{j,r},C_s)}{N(X_r,x_{j,r})} \right|^2}$$

$$\text{norm-vdm3}(x_{i,r},x_{j,r}) = \sqrt{k \times \sum_{s=1}^{k} \left| \frac{N(X_r,x_{i,r},C_s)}{N(X_r,x_{i,r})} - \frac{N(X_r,x_{j,r},C_s)}{N(X_r,x_{j,r})} \right|^2}$$

Calculando Similaridade Numérica

- Tradicionalmente no intervalo [0.0, 1.0]:
  - 0.0 = sem similaridade, 1.0 = identidade
- Similaridade = 1.0 - distância

Calculando Similaridade Booleana/Categórica

- Dados dois vetores booleanos X e Y, seja A o número de atributos onde ambos vetores assumem 1, etc. como mostrado abaixo
- Dois métodos para similaridade são dados ao lado
- Podem ser generalizados para dados categóricos

		Y[j] 1	Y[j] 0
X[j]	1	A	B
	0	C	D

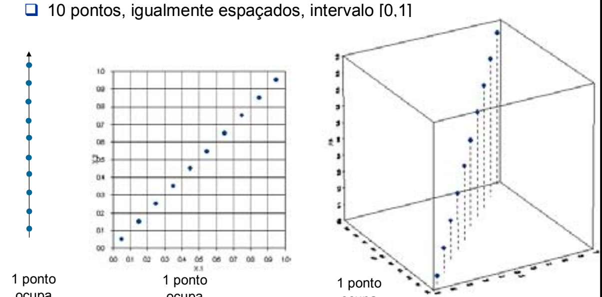
- Correlação = (A+D)/(A+B+C+D)
- Coef. Jaccard = A / (A+B+C+D)
  - Utilizado quando a ausência de um valor verdadeiro não significa similaridade
  - **Exemplo:**
    - Suponha que estamos realizando um trabalho de filogenética estrutural e X[j] é verdadeiro se o organismo tem asas
    - Dois organismos não são mais similares se ambos não têm asas
    - Dessa forma, o coeficiente de Jaccard é mais natural que o coeficiente de correlação neste caso

Maldição da Dimensionalidade

- O número de pontos necessários para manter uma determinada precisão (densidade espacial) cresce exponencialmente com o aumento na dimensão (acréscimo de novos atributos)
- Alternativamente, em altas dimensões, os pontos tendem a ser tornar eqüidistantes uns dos outros

Maldição da Dimensionalidade

- 10 pontos, igualmente espaçados, intervalo [0,1]



1 ponto ocupa 10% espaço

1 ponto ocupa 1% espaço

1 ponto ocupa 0.1% espaço

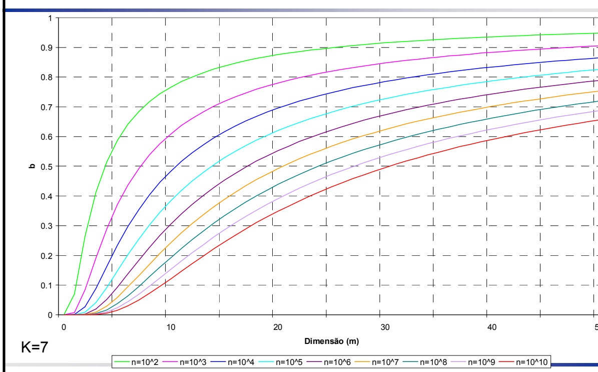
Maldição da Dimensionalidade

- Cada atributo adicional aumenta uma dimensão, assumindo não mais de 10 valores possíveis para cada atributo:
  - 2 atributos, ~100 combinações
  - 3 atributos, ~1,000 combinações
  - 4 atributos, ~10.000 combinações
  - 30 atributos…
  - 10.000.000.000.000.000.000.000.000.000.000 combinações

Maldição da Dimensionalidade

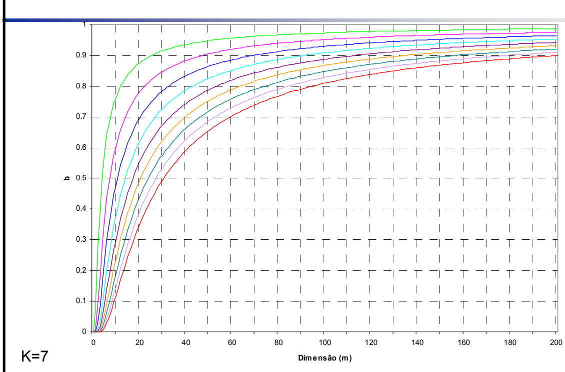
- Os vizinhos mais próximos em espaços de altas dimensões normalmente estão muito distantes entre si
- Em outras palavras, um ponto no espaço encontra-se quase na mesma distância de todos os demais pontos
- Considere um conjunto de **n** exemplos num espaço hipercúbico com **m** dimensões e suponha vizinhanças hipercúbicas de lado **b** e volume b^m
- Para conter **K** pontos, a vizinhança deve ocupar uma fração **K/n** do volume total que é 1
  - n (pontos) → 1 (volume)
  - K (pontos) → b^m (volume)
- Assim $b^m = K/n$ ou $b = (K/n)^{1/m}$
  - Assumindo K=7, m=100 e n=1.000.000 então $b \cong 0.888$
  - Isto significa que a vizinhança tem que abranger quase todo os espaço (quase 89%)

Maldição da Dimensionalidade



K=7

Maldição da Dimensionalidade



K=7

Resumo

- Métodos IBL baseiam suas decisões na similaridade de exemplos conhecidos ao invés de construir abstrações
- Conseqüentemente, eles têm tempo pequeno de aprendizado, mas um grande tempo de classificação (*lazy*)
- Vantagens
  - Não requer nenhum mecanismo de raciocínio
  - Não se baseia em abstrações de conceitos
  - É capaz de modelar fronteiras (problemas/conceitos) complexas

Resumo

- Desvantagens
  - Necessidade de definir similaridade métrica para objetos no universo
    - Métricas apropriadas de distância
  - Representação não é inteligível aos humanos
  - Para a classificação de um novo exemplo, todos os dados de treinamento devem estar disponíveis
  - Computacionalmente dispendioso em altas dimensões
    - Indexação eficiente dos exemplos de treinamento
  - A métrica de distância pode se tornar enganadora se todos os atributos são considerados
    - Em altas dimensões, todos os pontos encontram-se quase na mesma distância entre si
    - Tratamento para atributos irrelevantes
  - (K-NN) Escolha de K é desconhecida
    - uso de *cross-validation* para determinar K