Tema 3

Redução de Dimensionalidade

Seleção de Características

(e medidas de distância para classificação)

Professora:
Ariane Machado Lima

Seleção de Características

- Dados com N características
- Problema: escolher um subconjunto contendo D características que seja melhor que os outros subconjuntos (D < N)
- Número de possíveis subconjuntos de tamanho D:

Seleção de Características

- Dados com N características
- Problema: escolher um subconjunto contendo D características que seja melhor que os outros subconjuntos (D < N)
- Número de possíveis subconjuntos de tamanho D:

$$\left| \begin{matrix} N \\ D \end{matrix} \right| = \frac{N!}{(N-D)!\ D!}$$

- Número de possíveis subconjuntos: 2^N

Seleção de Características

- N = 20, D = 10: $2x10^5$ subconjuntos!

Seleção de Características

- N = 20, D = 10: $2x10^5$ subconjuntos!
- N = 40, D = 10: $\sim 10^9$ subconjuntos!

Seleção de Características

- N = 20, D = 10: $2x10^5$ subconjuntos!
- N = 40, D = 10: $\sim 10^9$ subconjuntos!

- N = 40 e D não definido: $2^{40} \sim 10^{12}$

- Milagre (ou maldição) da combinatória!!!

Seleção de Características

- Questões relevantes:
  - Como encontrar o melhor subconjunto? Avaliar todos, um a um?
  - O que é um subconjunto ser melhor que outro?

Características de um algoritmo de seleção de Características

- Relação com o classificador
- Função de avaliação (define métrica do que é “melhor”)
- Forma de percorrer o espaço de busca (subconjuntos de características)
- Critério de parada: quando me dou por satisfeito

Características de um algoritmo de seleção de Características

- Relação com o classificador
- Função de avaliação (define métrica do que é “melhor”)
- Forma de percorrer o espaço de busca (subconjuntos de características)
- Critério de parada: quando me dou por satisfeito

Relação com o classificador

- Filtro:
  - seleção de características é um passo anterior e independente da classificação (não tem relação nenhuma com o classificador)
- Camada (*wrapper*):
  - o subconjunto de características é avaliado usando o próprio classificador (avaliação pelo desempenho da classificação)
- Embutidos:
  - A seleção é feita pelo próprio classificador

Características de um algoritmo de seleção de Características

- Relação com o classificador
- Função de avaliação (define métrica do que é "melhor")
- Forma de percorrer o espaço de busca (subconjuntos de características)
  - Critério de parada: quando me dou por satisfeito

Função de avaliação (função critério)

- Dentre vários subconjuntos de características, qual é o melhor?
- Otimalidade é sempre relativa a uma função de avaliação que faz alguma medida do subconjunto corrente.
- Tipos de medidas:
  - Distância
  - Informação
  - Dependência
  - Consistência
  - Taxa de erro do classificador

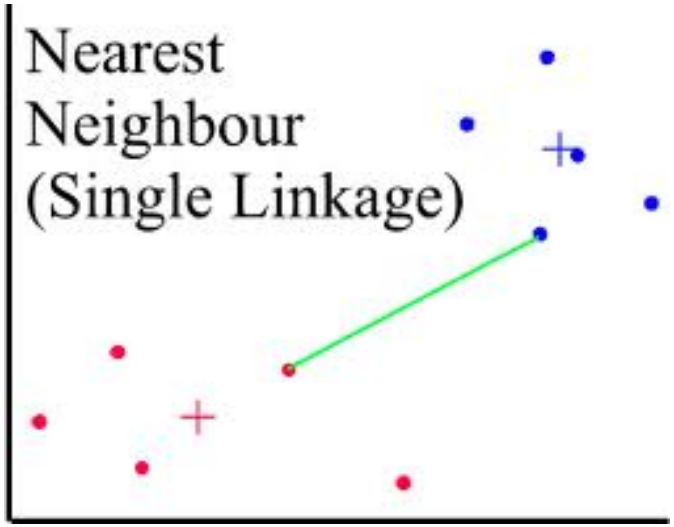
Função de avaliação (função critério)

- Medidas de distância
  - Ex: distância euclidiana: d(u,v) = ||v-u||
  - Subconjunto (de características) X é melhor do que o subconjunto Y se:

    X consegue fazer com que as duas classes fiquem mais distantes do que Y consegue
  - Qual é a distância entre duas classes?

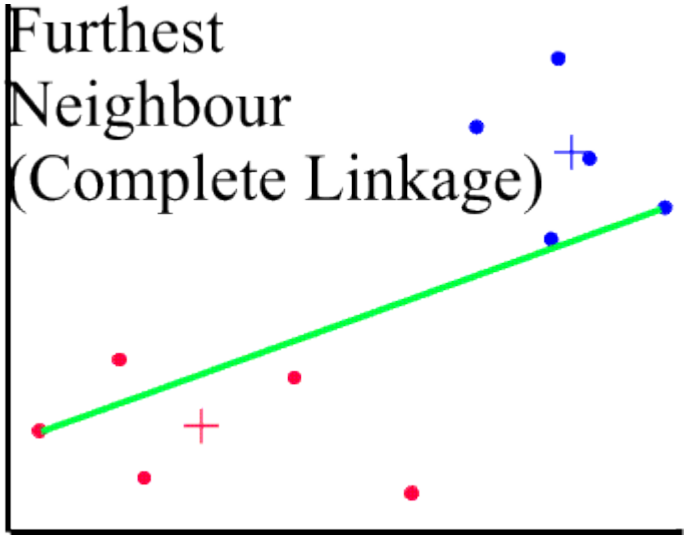
Distância entre duas classes A e B

- Single linkage (vizinhos mais próximos):
  - dist(A,B) = min dist(a,b), a ∈ A, b ∈ B
  - Distância mínima entre qualquer ponto da classe A e qualquer ponto da classe B



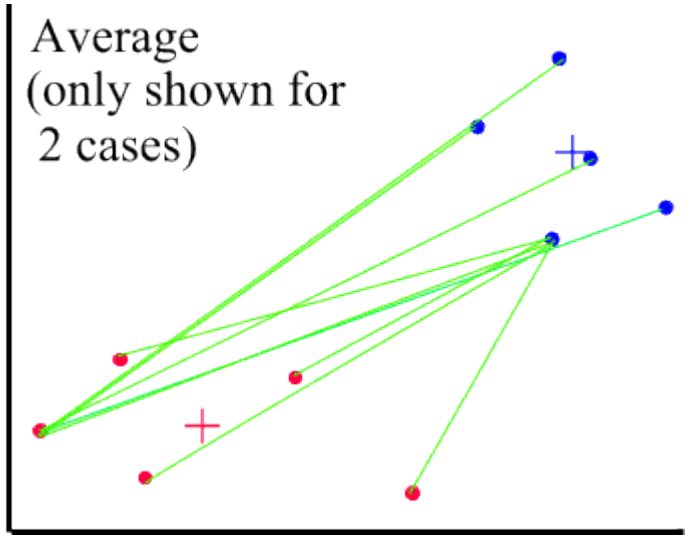
Distância entre duas classes A e B

- Complete linkage (vizinhos mais distantes):
  - dist(A,B) = max dist(a,b), a ∈ A, b ∈ B
  - Distância máxima entre qualquer ponto da classe A e qualquer ponto da classe B



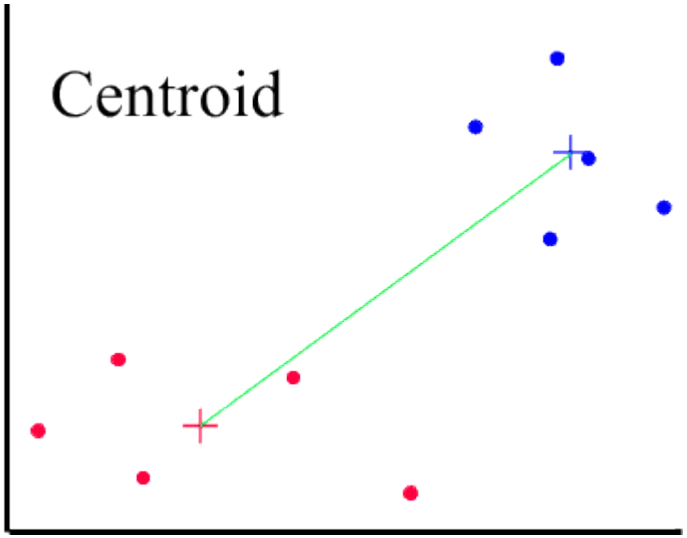
Distância entre duas classes A e B

- Distância média:
  - $dist(A,B) = 1/(N_A N_B) \sum dist(a,b)$, $a \in A$, $b \in B$
  - Distância média de todos os pares de pontos das classes A e B



Distância entre duas classes A e B

- Distância entre centróides:
  - dist(A,B) = dist (C_A,C_B)
  - Distância entre os centros de massa (centróides) da classe A (C_A) e da classe B (C_B)
  - I-ésimo componente do centróide da classe k (C^i_K) é valor médio dos i-ésimos componentes de todos os elementos (vetores de características) da classe k



Função de avaliação (função critério)

- Medidas de informação
  - Ex: informação mútua, entropia

Função de avaliação (função critério)

- Medidas de informação
  - Ex: informação mútua, entropia
  - Subconjunto X é melhor que Y se
    a informação mútua entre X e a classe é maior que a informação mútua entre Y e a classe
    (ie, olhando para X você acerta mais a classe do que olhando Y)

Função de avaliação (função critério)

- Medidas de informação
  - Ex: informação mútua, entropia

    Subconjunto X é melhor que Y se (entropia):

    a incerteza (entropia) sobre a classe diminui mais olhando para X do que para Y

    (ie, X fornece mais informação sobre a classe do que Y)

Função de avaliação (função critério)

- Medidas de dependência
  - Ex: correlação

    Subconjunto X é melhor que Y se:

    correlação (X, classe) > correlação(Y, classe)

    ou

    correlação(X, Y) é alta, um deles pode ser descartado

Função de avaliação (função critério)

- Medidas de consistência

  Ex: Subconjunto X é consistente se todas as instâncias com o mesmo valor de X possuem a mesma classificação

Função de avaliação (função critério)

- Medidas baseadas na taxa de erro do classificador
  - Subconjunto A de características é melhor que o subconjunto B se o classificador apresentou menor erro considerando A do que considerando B
  - Relação com o classificador: *wrapper*

- Veremos na próxima aula diferentes métodos para medir esse erro

Avaliação das funções de avaliação

- Generalidade
  - O subconjunto selecionado é bom para vários classificadores?
- Complexidade de tempo
  - Tempo gasto para selecionar o subconjunto
- Acurácia
  - Acurácia da classificação usando o subconjunto selecionado

Avaliação das funções de avaliação

Medida	Generalidade	Complex. Tempo	Acurácia
Distância	Sim	Baixa	-
Informação	Sim	Baixa	-
Dependência	Sim	Baixa	-
Consistência	Sim	Moderada	-
Taxa de erro do classificador	Não	Alta	Alta

- : nada se pode afirmar

Características de um algoritmo de seleção de Características

- Relação com o classificador
- Função de avaliação (define métrica do que é "melhor")
- Forma de percorrer o espaço de busca (subconjuntos de características)
- Critério de parada: quando me dou por satisfeito

Forma de percorrer o espaço de subconjuntos de características

- Completo:
- Heurístico:
- Randômico:

Forma de percorrer o espaço de subconjuntos de características

- Completo: encontra a solução ótima
  - Exaustivo (testa todos os subconjuntos)
  - Busca + backtracking (nem todos são avaliados)
    - Ex: branch and bound, best first search, beam search
- Heurístico:
- Randômico:

Forma de percorrer o espaço de subconjuntos de características

- Completo: encontra a solução ótima
  - Exaustivo (testa todos os subconjuntos)
  - Busca + backtracking (nem todos são avaliados)
    - Ex: branch and bound, best first search, beam search
- Heurístico: incremental ou baseado em pesos
- Randômico:

Forma de percorrer o espaço de subconjuntos de características

- Completo: encontra a solução ótima
  - Exaustivo
  - Busca + backtracking (nem todos são avaliados)
    - Ex: branch and bound, best first search, beam search
- Heurístico: incremental ou baseado em pesos
- Randômico: cada novo subconjunto é gerado aleatoriamente.
  - Definição de um número máximo de iterações

Branch and bound (completo)

- Quero selecionar D características de um total de N
- Crio uma árvore onde
  - Cada nó possui um subconjunto de características
  - A raiz possui o subconjunto (não próprio) contendo TODAS (N) as características
  - Um nó filho tem uma característica a menos que seu pai
  - Folhas têm subconjuntos com D características
- Busca em profundidade para achar a folha melhor avaliada

Branch and bound

Árvore

Branch and bound

Entrada:

- uma função critério M()
- uma amostra de treinamento (rotulada)
- D (número desejado de características)
- a árvore de características ($T_i(n)$ é o i-ésimo nó do nível n; nível contado de D a N (das folhas até a raiz)

1. melhor_medida ← 0
2. S ← ∅ // subconjunto de característica
3. Explore_no($T_1(N)$)

Saída: S terá o subconjunto melhor avaliado

Branch and bound

- A função de avaliação utilizada deve ser monotonicamente crescente na dimensão (nr de características), isto é, para subconjuntos A e B, M(A) <= M(AUB) ,

isto é, se eu tirar elementos de um subconjunto, a medida de avaliação do subconjunto resultante não pode aumentar (fica igual ou piora)

- Ideia: Se eu já sei que um subconjunto de tamanho D tem uma medida m1, não vou analisar subconjuntos maiores com medidas <= m1, pois esse valor só pode piorar ao remover algumas características

Explore_no($T_i(n)$):

Se ($M(T_i(n)) \leq$ melhor_medida) *retorna*; // se o nó não é promissor, não siga sua subárvore, pois tirar características desse subconjunto não vai aumentar M()

Se (n = D) então // $M(T_i(n))$ > melhor_medida, senão teria retornado acima

melhor_medida ← $M(T_i(n))$

S ← $T_i(n)$

retorna *//ie, melhor_medida é sempre de uma folha (com exceção da inicialização)*

// para um nó não-folha ser avaliado, ele precisa ter um M maior do que a da melhor folha avaliada até o momento

Para todos ($T_k(n-1) \subset T_i(n)$) *// filhos do nó $T_i(n)$*

Explore_no($T_k(n-1)$)

retorna

Branch and bound

- Observações
  - A função de avaliação utilizada deve ser monotonicamente crescente na dimensão
    - Melhor usar com funções de avaliação baseadas em distância
  - Encontra a solução ótima
  - Apesar de não exaustivo, ainda pode ser computacionalmente intenso (dependendo de N e D)

Heurísticos

- Weighting: durante a execução as características recebem pesos; no final são selecionadas as que têm peso maior que um limiar
- Forward selection: a cada iteração novas características são adicionadas ao subconjunto corrente
  - Ex: sequential forward selection (SFS)
- Backward selection: a cada iteração características são deletadas do subconjunto corrente
  - Ex: sequential backward selection (SBS)
- Forward-Backward: combinação de ambos
  - Ex: plus t – take away r

Heurísticos – Sequential Forward Selection

- Início: conjunto vazio (raiz da árvore)
- A cada iteração: testa o novo subconjunto composto pelo subconjunto corrente + uma de cada característica faltante. O novo subconjunto corrente será o melhor avaliado
- Continua até alcançar D características, ou o melhor de todos

Heurísticos – Sequential Forward Selection

- Início: conjunto vazio (raiz da árvore)
- A cada iteração: testa o novo subconjunto composto pelo subconjunto corrente + uma de cada característica faltante. O novo subconjunto corrente será o melhor avaliado
- Continua até alcançar D características
- Problema:

Heurísticos – Sequential Forward Selection

- Início: conjunto vazio (raiz da árvore)
- A cada iteração: testa o novo subconjunto composto pelo subconjunto corrente + uma de cada característica faltante. O novo subconjunto corrente será o melhor avaliado
- Continua até alcançar D características
- Problema: depois que uma característica entra, não sai mais

Heurísticos – Sequential Backward Selection

- Início: conjunto de N características (raiz da árvore)
- A cada iteração: testa o novo subconjunto composto pelo subconjunto corrente – (menos) uma de cada característica. O novo subconjunto corrente será o melhor avaliado
- Continua até alcançar D características
- Figura (semelhante à da branch and bound)

Heurísticos – Sequential Backward Selection

- Início: conjunto de N características (raiz da árvore)
- A cada iteração: testa o novo subconjunto composto pelo subconjunto corrente – (menos) uma de cada característica. O novo subconjunto corrente será o melhor avaliado
- Continua até alcançar D características
- Figura (semelhante à da branch and bound)
- Problema:

Heurísticos – Sequential Backward Selection

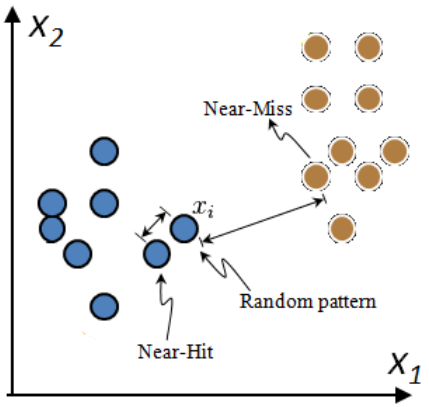
- Início: conjunto de N características (raiz da árvore)
- A cada iteração: testa o novo subconjunto composto pelo subconjunto corrente – (menos) uma de cada característica. O novo subconjunto corrente será o melhor avaliado
- Continua até alcançar D características
- Figura (semelhante à da branch and bound)
- Problema: depois que uma característica sai, não entra mais

Heurísticos – plus l – take away r

- Início: conjunto vazio (raiz da árvore)
- A cada iteração:
  - A) testa o novo subconjunto composto pelo subconjunto corrente + l novas características. O novo subconjunto corrente será o melhor avaliado
  - B) testa o novo subconjunto composto pelo subconjunto corrente – r características. O novo subconjunto corrente será o melhor avaliado

Heurísticos –Weighting

- Ex: RELIEF (distância euclidiana):
- Para um certo número de instâncias (NR_AMOSTRA) são encontrados:
  - NearHit (instância mais próxima da mesma classe)
  - NearMiss (instância mais próxima de classe diferente)



Heurísticos –Weighting

- Ex: RELIEF (distância euclidiana):
- Para um certo número de instâncias (NR_AMOSTRA) são encontrados:
  - NearHit (instância mais próxima da mesma classe)
  - NearMiss (instância mais próxima de classe diferente)
- As características têm seus pesos atualizados, sendo consideradas:
  - Mais relevantes se distinguem a instância de seu nearMiss
  - Menos relevantes se distinguem a instância de seu nearHit

Heurísticos –Weighting

Relief (Dataset, NR_AMOSTRA, PESO_MIN):

S = vazio //características selecionadas

Para j =1 a N $W_j = 0$ //pesos de cada uma das N características f_j's

Para i =1 a NR_AMOSTRA

    Escolha aleatoriamente uma instância x no Dataset

    Encontre seus NearHit e NearMiss

    Para cada característica j = 1 a N

        $W_j = W_j$ – $diff(x_j, NearHit_j)^2 + diff(x_j, NearMiss_j)^2$

Para j = 1 a N

    Se W_j >= PESO_MIN S = S “+” f_j

Retorna S

Heurísticos –Weighting

diff (x_j, y_j):

Se f_j é uma variável nomimal:

diff(x_j, y_j) = 0 se $x_j = y_j$

1 caso contrário

Se f_j é uma variável numérica:

diff(x_j, y_j) = $(x_j - y_j) / n_j$

no qual n_j é um valor para normalizar f_j em [0,1]

Heurísticos –Weighting

Atualmente há novas versões do Relief, como o Relief-F, que usa os K nearHits e K nearMisses.

Artigo no TIDIA.

Resultados não tão bons quanto outros em um trabalho específico de classificação TEA x não TEA, utilizando distâncias de componentes faciais.

Provavelmente por conta de

Heurísticos –Weighting

Atualmente há novas versões do Relief, como o Relief-F, que usa os K nearHits e K nearMisses.

Artigo no TIDIA.

Resultados não tão bons quanto outros em um trabalho específico de classificação TEA x não TEA, utilizando distâncias de componentes faciais.

Provavelmente por conta de não levar em consideração o efeito COMBINADO de cada subconjunto de características, e sim cada uma individualmente.

Randômicos

- Cada novo subconjunto é gerado aleatoriamente
- Exemplos:
  - LVF – Las Vegas Filter
  - Baseados em algoritmo genético

Randômico – LVF (Las Vegas Filter)

Avalia um número pré-determinado de subconjuntos de características (MAX_TENTATIVAS)

- Cada subconjunto **S** é escolhido aleatoriamente (tamanho e elementos)
- Há a melhor solução corrente (**Sbest**)
  - **Cbest** é a cardinalidade do subconjunto Sbest
- O próximo subconjunto só é analisado se sua cardinalidade for menor ou igual a Cbest
  - Se sua análise for boa (taxa de inconsistência menor ou igual a um limiar), torna-se Sbest

Randômico – LVF (Las Vegas Filter)

Entrada:

MAX_TENTATIVAS

D (dataset)

N (número total de características)

L (limiar de taxa de inconsistência)

Saída:

Subconjuntos (vários) que satisfazem o critério de inconsistência

Randômico – LVF (Las Vegas Filter)

- Critério de inconsistência (S, D)
  - Duas instâncias são **inconsistentes** se suas características de S são iguais mas suas classes não
  - Considerando que existam q conjuntos de instâncias inconsistentes (cada conjunto contendo as instâncias com os mesmos valores das características S). Para cada conjunto:
    - n instâncias inconsistentes = $c_1 + c_2 + \ldots$ (onde c_i é o número de instâncias da classe i neste conjunto)
    - O **contador de inconsistência** deste conjunto = $n\text{-}c_{max}$, onde $c_{max} = \max(c_1, c_2, \ldots)$
  - **Taxa de inconsistência** = soma de todos os contadores de inconsistência / | D |
- Observação: não apropriado para valores contínuos

Randômico – LVF (Las Vegas Filter)

```
Cbest = N
Para i = 1 até MAX_TENTATIVAS
    S=sorteiaSubconjunto(semente)
    C = cardinalidade(S)
    se (C ≤ Cbest)
        se (taxaInconsistencia(S, D) <= L)
            imprime (C, S)
            se (C < Cbest)
                Cbest = C
```

Randômico – LVF (Las Vegas Filter)

- Limiar L:
  - Valor default: L
  - Pode ser = taxaInconsistência (S^A, D), onde S^A é o conjunto de todas as características

Randômico – LVF (Las Vegas Filter)

Critério de parada?

- número máximo de tentativas (sugestão experimental: 77*N)
- Balanço entre tempo e qualidade da solução
- Na verdade pode-se parar a qualquer momento (e escolher o último subconjunto impresso)

Randômico – Algoritmo Genético

- População de indivíduos (tamanho constante): amostra do espaço de busca (de soluções)
- Cada indivíduo é avaliado por uma função fitness
- Os indivíduos mais aptos (com melhor avaliação) geram descendentes (que herdam parte de suas características)
  - Crossover: mistura de 2 pais (origem a 2 filhos)
  - Mutação: alteração de 1 pai
- Os indivíduos menos aptos morrem
- MAX_GERACOES: número de gerações que são produzidas (critério de parada)

Randômico – Algoritmo Genético para Seleção de Características [VAFAIE & IMAM, 1994]

- Indivíduo: vetor binário de tamanho n (0 na i-ésima posição indica a ausência da característica i, e 1 sua presença)
- Crossover: dada uma posição, concatena a primeira parte de um com a segunda parte de outro (gerando 2 filhos distintos)
- Mutação: altera aleatoriamente um ou mais componentes de um pai
- Função fitness: desempenho do classificador (wrapper)

Em resumo...

- Cada algoritmo de seleção de características pode ser entendido a partir dessas informações:
  - É do tipo filtro, camada ou embutido
  - Qual função critério é utilizada? (pode ser a combinação de uma ou mais
  - Como ele percorre o espaço de busca?
    - Acha o ótimo dentre todas as combinações possíveis? (logo, a função critério deve avaliar o subconjunto como um todo)
    - Usa alguma heurística para escolher um subconjunto? (selecionando heuristicamente cada subconjunto a ser avaliado, ou dando uma nota para cada característica separadamente e arbitrariamente escolhendo as melhores)
    - De forma aleatória

Exemplos

- **CFS: Correlation-based feature selection** - características altamente correlacionadas com a classe alvo, porém pouco correlacionadas entre si (HALL, 1990)
- **mRMR: Minimum Redundance Maximum Relevance** - dois critérios: mínima redundância entre pares de caracterísssticas e relevância máxima por meio da medida de informação mútua (DING; PENG, 2005)

Trabalho

Seleção de características usando dois métodos à escolha (de preferência o mais distintos possível)

Métodos: uso de quais rotinas, parâmetros utilizados e por quê, outras informações relevantes

Por ex: no caso do Relief, como selecionou as características com base nos pesos e por quê

Resultados: quantas e quais características foram selecionadas

Discussão: o que você achou dos resultados? Reduziu bastante?

Observações sobre o trabalho

- Precisa usar uma coluna de classificação.

- No caso do Relief: Função cutoff (do mesmo pacote) para selecionar o subconjunto de características

Referências

- DASH, M., LIU, H., Feature Selection for Classification. Intelligent Data Analysis 1, 131-156, 1997.
- DING, C.; PENG, H. Minimum redundancy feature selection from microarray gene expression data. Journal of bioinformatics and computational biology, World Scientic, v.3, n. 02, p. 185-205, 2005.
- HALL, M. A. Correlation-based feature selection for machine learning. University of Waikato Hamilton, 1999.
- LIU, H., SETIONO, R., A Probabilistic Approach to Feature Selection - A Filter Solution. In: Proceedings of International Conference on Machine Learning, 319-327, 1996. (alg. LVF)
- VALFAIE, H., IMAM, I.F., Feature selection methods: genetic algorithms vs. greedy-like search. In: Proceedings of the International Conference on Fuzzy and Intelligent Control Systems, 1994