UNIVERSIDADE FEDERAL DE PERNAMBUCO
GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO
CENTRO DE INFORMÁTICA

UM ESTUDO SOBRE FUNÇÕES DE DISTÂNCIA APLICADAS A ALGORITMOS DE APRENDIZAGEM DE MÁQUINA

TRABALHO DE GRADUAÇÃO

Aluno: Tiago Buarque Assunção de Carvalho (tbac@cin.ufpe.br)
Orientador: George Darmiton da Cunha Cavalcanti (gdcc@cin.ufpe.br)
Co-orientador: Tsang Ing Ren (tir@cin.ufpe.br)

AGOSTO DE 2007

Mal-Estar Tecnológico

Quando as máquinas souberem
o que estou pensando
se tornará inconveniente
o contato com outro ser humano

Pois o que é tão trabalhoso
será ainda menos vantajoso
pela utilidade que terá

E se era tão deprimente
procurar um semelhante
para ser compreendido

Venderão
a preços populares
a melhor esposa
e o melhor amigo

Não padecerei da dúvida
de saber o que eu gosto
Com a sublime facilidade
de ter sempre pai e mãe a postos

Jamais sentirei a alegria
de ser conhecido pela minha companheira
sabendo que ela nunca me compreenderá
tão bem como a minha cafeteira.

Tiago Buarque (2007)

Agradecimento

Agradeço primeiro a Deus, que estourou o *Big Bang*, na seqüência, agradeço a meus avós e seus ancestrais a partir dos mais longínquos, por terem escolhido viver e me permitir fazer a mesma opção. Agora agradeço aos meus pais, agentes curadores do meu crescimento: que me alimentaram, vestiram, obrigaram a ir ao colégio e ensinaram-me valores morais. Às infindáveis listas de exercício de Matemática que mamãe passava para mim na infância. E à perspectiva mercadológica – onde se embutia a necessidade de manutenção da vida – que papai teve ao me tirar a dúvida sobre que curso fazer.

Agradeço às minhas duas irmãs, tão queridas e carinhosas não obstante as discussões – fruto da intimidade. Ao meu grandessíssimo amigo Filipe, amigo das antigas, amigo das antrolas, há 22 anos de alegrias e contendas. E todos os outros elementos que nessa misteriosa vida me proporcionaram qualquer experiência social, partes transformadoras de todo o meu modelo metal.

Agradeço a Sid Clapis e ao Saulo Souto que me proporcionaram companhia e conversas fundamentais para descobrir o sentido da vida durante as madrugadas de escrita deste relatório. Por último agradeço ao amigo Jera que me ajudou a fazer a capa.

Resumo

É imensa a listas de algoritmos na área de Aprendizagem de Máquina que utilizam funções de distância. Podemos citar os algoritmos de agrupamento como o k-meas, as redes neurais Kohonen e RBF, além do exemplo mais claro que é o k-NN. O objetivo desse trabalho é mostrar que diferentes funções de distâncias podem ser empregadas para melhorar a performance de tais algoritmos. Utilizaremos como estudo de caso o k-NN e a rede RBF treinada com DDA.

Uma vez que a forma como se realiza essa medida pode interferir no comportamento do algoritmo, grande variedade de funções de distância foi desenvolvida justamente para se conseguir melhores resultados nos algoritmos que as empregam. Recentemente [Wang 06] propôs uma nova função de distância baseado no conceito de vizinhanças, o que mostra que o cálculo de distâncias não é um problema completamente resolvido. Tal dificuldade emergiu com surgimento de novos e complexos tipos de dados.

Esse trabalho compara o comportamento das funções avaliadas por Wang, HEOM, HVDM, DVDM, IVDM, NCM e mais algumas variações dessa última aqui propostas. Para tanto essas funções serão utilizadas em dois algoritmos de classificação: o k-NN e as redes RBF. Também é testada a resposta às modificações desses algoritmos sob as mesmas funções de distância.

Índice

1. Introdução

Aprendizagem Baseada em Instâncias (*Instance Based Learning* – IBL) é um paradigma de aprendizagem no qual os algoritmos tipicamente guardam alguns ou todos os *n* padrões de treinamento disponíveis uma base de dados. Os elementos ou instâncias dessa base são padrões compostos por um vetor de atributos e uma classe. Os algoritmos IBL generalizam (classificam) um novo padrão apresentado associando ele a um ou mais elementos do conjunto de treinamento. Uma forma de realizar essa associação é por meio da distância entre o padrão que se pretende classificar e os elementos da base. Essa abordagem se difere de outras pois a generalização é feita no momento da classificação e não durante o treinamento.

k-vizinhos mais próximos (*k-Nearest Neighbors* ou k-NN), *Locally Weighted Regression* (regressão com pesos locais), Redes Neurais RBF e *Case-Based Reasoning* (Raciocínio Baseado em Casos) são alguns tipos de algoritmos IBL [Mitchell 97]. O k-NN e as redes RBF têm a característica de usarem funções de distância no centro do seus algoritmos. A rede RBF treinada com DDA, que é a utilizada nesse trabalho, cria um modelo durante a sua fase de treinamento, contudo alguns dos padrões de treinamento ainda são armazenados e utilizados na fase de generalização.

Contudo calcular a distância entre dois elementos de uma base de dados composta de dados medidos no mundo real nem sempre é uma tarefa trivial. Uma vez que os atributos do vetor de características podem ser de vários tipos, em particular destacamos dois desses tipos: numérico e categórico. O tipo numérico é um número contínuo ou discreto. O categórico é um nome ou um rótulo.

Recentemente [Wang 06] propôs NCM (*Neighborhood Counting Measure*), uma nova forma de calcular distâncias entre esses vetores de atributos mistos (onde os atributos podem se categóricos ou numéricos) Aplicou essa função de distância ao k-NN e comparou os resultados para o mesmo algoritmo utilizando outras funções de distâncias listadas em [Wilson; Martinez 97] e mostra que para cada base de dados existe uma função de distância que melhor se adequa.

Esse trabalho compara o comportamento das funções avaliadas por Wang, HEOM, HVDM, DVDM, IVDM, NCM e mais algumas variações dessa última aqui propostas. Para tanto essas funções serão utilizadas em dois algoritmos de classificação: o k-NN e as redes RBF. Também é testada a resposta às modificações desses algoritmos sob as mesmas funções de distância.

Esse trabalho de graduação está organizado da seguinte forma: no capítulo 2 vemos os problemas e soluções para a codificação dos dados, algumas formas do cálculo de distância amplamente conhecidas na literatura e NCM (*Neighborhood Counting Measure*) uma métrica bastante recente. No capítulo 3 vamos rever alguns desses algoritmos de aprendizagem que utilizam funções de distância como um dos seus principais componentes. Os testes desses algoritmos com as funções de distâncias estão no capítulo 4. No último capítulo estão as considerações finais e propostas de trabalhos futuros.

2. Distância

Agora vamos introduzir os conceitos necessários e as funções para o cálculo de distância. Essas serão as funções utilizadas nos algoritmos do capítulo 3. Primeiro vemos os tipos de dados sobre os quais trabalhamos. Na seqüência como calcular a distância entre esses dados.

Entre os tipos de dados destacamos os numérico e os categóricos. Nosso objetivo é definir funções de distâncias que trabalhem com ambos os dados. Analisaremos primeiramente distâncias que operam somente sobre atributos numéricos ou somente sobre categóricos depois chegamos às funções heterogêneas: HEOM, HVDM, DVDM, IVDM e NCM.

2.1. Tipos de Dados

Um objeto de uma base de dados é representado por um vetor de características. À construção desse vetor dá-se o nome de "extração de características". Cada característica é mapeada em um atributo. Por isso podemos dizer que a representação de um elemento é dada por meio de um vetor de atributos.

Os tipos desses atributos podem ser diferentes. Cada tipo permite certas operações que dão uma resposta numérica. Por meio dessas operações sobre os atributos é que se realiza o cálculo das distâncias.

Estão descritos em [Wang 06], [Healey 90] quatro tipos de atributos: nominal, ordinal, intervalar e racional. Atributos **nominais** também chamados **categóricos** são o tipo mais básico de atributo, pois podem descrever qualquer coisa, são rótulos, nomes ou códigos que representam valores, e só podem ser comparados se são iguais ou diferentes. Quando uma variável é medida nominalmente suas categorias devem ser mutuamente excludentes e exaustivas. Atributos **ordinais** são também categóricos, porém podem ser ordenados, isto é permitem a operação de "maior que". Atributos

intervalares possuem a mesmas características dos ordinais, contudo nesse tipo as operações de adição e subtração passam a fazer sentido, uma vez que os valores são separados pelo mesmo intervalo. Atributos do tipo **racional**, aqui também chamado **numérico**, possuem as propriedades dos intervalares além de existir razão entre os seus valores, de tal forma são permitidas operações de multiplicação e divisão e, por conseguinte, várias outras, como raiz quadrada, logaritmo etc. O valor zero na escala racional não é arbitrário diferentemente do permitido na escala intervalar, por causa da operação de multiplicação permitida em uma e não na outra.

Podemos também definir um domínio para a base de dados através dos tipos de seus atributos. Dois elementos do mesmo domínio possuem o mesmo número n de atributos, assim, seja a uma posição no vetor de atributos de um elemento qualquer, $1 \leq a \leq n$, se dois elementos pertencem ao mesmo domínio os atributos da posição a de cada elemento têm o mesmo tipo. Uma vez que tratamos com conjuntos finitos de dados, podemos considerar, sem perda de generalidade, que se o atributo da posição a é do tipo categórico ele pertence a um conjunto finito que podemos chamar domínio de a ou $dom(a)$. E se o atributo a for numérico ele assumirá um valor máximo, $\max(a)$, e um valor mínimo, $\min(a)$. Nesse trabalho todas as funções de distância tratam apenas atributos do tipo categórico ou numérico. Munidos dessas definições podemos agora definir cálculos de distâncias entre dois elementos, isto é, entre dois vetores de atributos.

2.2 Entre vetores de atributos numéricos

Distâncias entre vetores de atributos puramente numéricos são mais comuns na literatura, o exemplo mais clássico que pode ser citado é a Distância Euclidiana[1]:

$$E(x, y) = \sqrt{\sum_{a=1}^{n} (x_a - y_a)^2}$$

[1] A Distância Euclidiana foi definida por volta de 300 a.C. e corresponde ao comprimento de um segmento e reta entre dois pontos num espaço euclidiano bi ou tri-dimensional. Essa distância euclidiana aqui apresentada é uma generalização para um espaço euclidiano n-dimensional.

onde *x* e *y* são dois vetores de *n* atributos numéricos. Assim $(x_a - y_a)$ é a diferença entre os atributos de *x* e *y* na posição a. A Distância Euclidiana é a raiz quadrada da soma do quadrado da diferença $(x_a - y_a)$, para todo *a*. Na prática a raiz quadrada não precisa ser computada, quando se usa o valor da distância apenas para fins de comparação.

Outra distância bastante conhecida é a *Manhattan* ou *city*-block que tem a vantagem de ter um custo computacional bem menor que a distancia Euclidiana. Essa e outras distâncias fora entre vetores de atributos numéricos foram levantadas por [Wilson; Martinez 97] e podem ser vistas na **Tabela 1**.

Um ponto que merece atenção é a normalização dos dados. Vejamos o exemplo: nossa base de treinamento está preenchida com elementos que se pretendem classificar com base na altura a e peso. Sejam duas instâncias a = {1,60; 50} e b = {1,70; 70}, onde 1,60m e 1,70m são as alturas de a e b respectivamente e 50kg e 60kg os pesos. A Distância Euclidiana entre esses dois elementos é:

$$E(a,b) = \sqrt{(1{,}70 - 1{,}60)^2 + (70 - 50)^2} = \sqrt{0{,}01 + 400{,}00} \cong 20$$

Percebe-se que a altura tem uma influencia desprezível no cálculo dessa distância. Para sanar essa limitação da distância faz-se necessária a normalização dos dados, que consistem em fazer com que os dados tenham a mesma ordem de grandeza. Por questão de simplicidade faremos com que os dados fiquem todos no intervalo [0;1].

Podemos definir a Distância Euclidiana Normalizada:

$$En(x,y) = \sqrt{\sum_{a=1}^{n} \left(\frac{|x_a - y_a|}{\max(a) - \min(a)} \right)^2}$$

onde max(a) e min(a) estão definidos no fim da seção 2.1.

<table>
<tr>
<td>Manhattan (city-block):
$D(x,y)=\sum_{a=1}^{n}|x_a-y_a|$</td>
<td rowspan="3">Correlação:
$D(x,y)=\frac{\sum_{a=1}^{n}(x_a-\bar{x}_a)(y_a-\bar{y}_a)}{\sqrt{\sum_{a=1}^{n}(x_a-\bar{x}_a)^2\sum_{a=1}^{n}(y_a-\bar{y}_a)^2}}$
$\bar{x}_a=\bar{y}_a$ é a média dos valores do atributo da posição a em todos os elementos do conjunto de treino.</td>
</tr>
<tr>
<td>Chebychev:
$D(x,y)=\max_{a=1}^{n}|x_a-y_a|$</td>
</tr>
<tr>
<td>Camberra:
$D(x,y)=\sum_{a=1}^{n}\frac{|x_a-y_a|}{|x_a+y_a|}$</td>
</tr>
<tr>
<td>Minkowsky:
$D(x,y)=\left(\sum_{a=1}^{n}|x_a-y_a|^r\right)^{\frac{1}{r}}$</td>
<td rowspan="2">Chi-quadrado:
$D(x,y)=\sum_{a=1}^{n}\frac{1}{soma_a}\left(\frac{x_a}{tam_x}-\frac{y_a}{tam_y}\right)^2$
soma_a é a soma de todos os calores do atributo a na base de dados e tam_x é a soma de todos os valores no vetor x.</td>
</tr>
<tr>
<td>Quadrática:
$D(x,y)=(x-y)^T Q(x-y)$
Q é a matriz de pesos $n\times n$.</td>
</tr>
<tr>
<td>Mahalanobis:
$D(x,y)=[\det V]^{1/n}(x-y)^T V^{-1}(x-y)$
V é a matriz de covariância de A1..An, e Aj é o vetor dos valores dos atributos da posição j de todos os elementos do conjunto de treino.</td>
<td>Correlação de Kendall:
$D(x,y)=\frac{2}{t(t-1)}\sum_{a=1}^{n}\sum_{a'=1}^{a-1}\text{sinal}(x_a-x_{a'})\,\text{sinal}(y_a-y_{a'})$
$\text{sinal}(x)=\begin{cases}-1, se\ x<0\\0, se\ x=0\\1, se\ x>0\end{cases}$
onde t é o número de elementos do conjunto de treino.</td>
</tr>
</table>

Tabela 1. Funções de distância entre vetores de atributos numéricos.

2.3 Entre de atributos categóricos

Distancia de Hamming

Para atributos categóricos a Distancia de Hamming, também conhecida com *overlap*, é uma distância muito utilizada e bastante simples. A Distancia de Hamming aplicada a dois atributos numéricos tem o valor 0, se esses atributos são iguais, ou 1, se esses atributos são diferentes.

$$h_a(x_a, y_a) = \begin{cases} 1, & se\ x_a \neq y_a \\ 0, & se\ x_a = y_a \end{cases}$$

Essa distância aplicada a vetores de atributos puramente categóricos se traduz como o número de valores diferentes nas mesmas posições. A distância de Hamming entre dois vetores de atributos categóricos é definida como:

$$H(x, y) = \sum_{a=1}^{n} h_a(x_a, y_a)$$

VDM – Value Difference Metric

A métrica VDM foi introduzida por [Stanfill; Waltz 86] *apud* [Wilson; Martinez 97] para prover uma função de distância mais apropriada entre atributos categóricos. Uma versão simples do VDM (sem esquema de pesos) entre dois atributos é dada por:

$$vdm_a(x, y) = \sum_{c=1}^{C} \left| \frac{N_{a,x,c}}{N_{a,x}} - \frac{N_{a,y,c}}{N_{a,y}} \right|^q = \sum_{c=1}^{C} \left| P_{a,x,c} - P_{a,y,c} \right|^q$$

Onde:

- $\boldsymbol{N_{a,x,c}}$ é o número de instâncias do conjunto de treinamento que pertencer à classe *c* e tem o valor *x* para o atributo *a*;

- $\boldsymbol{N_{a,x}}$ é o número de instâncias do conjunto de treinamento que tem o valor *x* para o atributo *a*; $N_{a,x}$ é a soma de $N_{a,x,c}$ para todo valor de *c*;

- ***C*** é o número de classes do problema;

- ***q*** é uma constante, usualmente 1 ou 2. O valor de *q* utilizado nesse trabalho será 2 por demonstrar melhores resultado como afirma [Wilson; Martinez 1997];

- $\boldsymbol{P_{a,x,c}} = P(c \mid x_a)$ é a probabilidade condicional de que a classe de um elemento ser *c* dado que o atributo *a* tem o valor x.

Baseado em $vdm_a(x,y)$ define-se VDM, uma função de distância entre dois vetores de atributos puramente categóricos:

$$VDM(x,y) = \sqrt{\sum_{a=1}^{n} vdm_a(x_a, y_a)}$$

Enquanto a distância de Hamming só verifica se dois atributos são iguais ou diferentes, vdm considera a distribuição entre as classes. De modo que se dois valores de atributos têm mesmo significado, mas rótulos diferentes, logo devem possuir a mesma distribuição entre as classes e responder com uma distância pequena no vdm, no caso de Hamming, por serem diferentes responderiam com a distancia máxima (1.0).

Para exemplificar vamos criar um exemplo fictício de um sistema identificador de fraude, cuja base contém atributos sobre pessoas submetidas às mesmas situações que fraudaram ou não fraudaram. Um atributo categórico dos elementos da base é religião. As classes possíveis na base de dados são duas: "fraudaria" e "não fraudaria" e o atributo religião pode assumir vários valores, e.g., "não tenho religião", "ateu", "católico", "protestante", "espírita", "candomblé", "pratico outra religião". Enfatizando que é um exemplo fictício, vamos assumir uma probabilidade – $P_{a,x,c}$ – mais alta de "fraudaria" para os valores "não tenho religião" e "ateu" do atributo religião e uma probabilidade mais baixa para "não fraudaria" quando a instância da base assume para o mesmo atributo os valores "católico", "protestante", "espírita", "candomblé" e "pratico outra religião". Ou seja, quem tem alguma religião apresente menor probabilidade de fraudar, por isso, estão mais próximos (menor distância) em relação a esse aspecto. Igualmente quem não tem religião ou é ateu também têm menor distância em relação ao aspecto de fraude. VDM possui uma robustez muito grande por causa disso, pois vê como duas instâncias estão próximas considerando a distribuição desses atributos nas classes. Enquanto Hamming só consideraria se os rótulos são diferentes, perdendo muita informação útil à classificação.

Um ponto deve ser chamado à atenção: quando o valor do atributo em questão nunca foi visto antes, ou seja, nenhuma instância da base possui aquele valor. Para Hamming é uma questão muito simples, pois só comparam se os atributos são diferentes. Porém para VDM não há qualquer informação sobre esse atributo. Se

considerarmos $N_{a,x,c} = 0$, também teremos $N_{a,x} = 0$ e por conseguinte $P_{a,x,c} = 0/0$, que é um valor indefinido. Nesse trabalho assume-se para esse caso $P_{a,x,c} = 0$, de modo que a distância desse atributo para qualquer outro da base será igual a 1, o que é a soma de $P_{a,y,c}$ para todas as classes com outro atributo que está sendo comparado. Ainda pode-se usar $P_{a,x,c} = 1/C$, onde C é o número de classes e a soma da $P_{a,x,c}$ vai ser igual a 1, o que é estatisticamente plausível. Ambas as formas de tratar esse caso são questões de decisão de projeto, foi escolhida a primeira, somente em experimentos poder-se-ia verificar qual das duas era mais adequada, possivelmente uma terceira forma de tratar esse caso ainda apresentasse melhores resultados. Contudo assume-se que esse problema em questão é raro, por tanto a forma de tratar isso não influenciará muito no desempenho do classificador.

2.4 Distâncias Heterogenias e Normalização

Conforme [Wilson; Martinez 97] problemas reais precisam tratar ao mesmo tempo de atributos categóricos e numéricos, citando como exemplo bases do *UCI Machine Learning Repository* [UCI 07]. Como, porém, calcular distâncias entre vetores de atributos mistos? Distâncias mistas, também chamadas heterogenias.

Recordamos da seção 2.1 que um elemento pode conter tipos distintos no seu vetor de atributos, desde que, para as mesmas posições do vetor, esses atributos sejam do mesmo tipo. Como visto anteriormente, a distância entre vetores é calculada como uma combinação das distâncias entre os atributos desses vetores. Do mesmo modo, se um vetor apresenta mais de um tipo de atributo, pode-se definir uma função de distância heterogênea como sendo a combinação de distâncias entre atributos, cada qual operando no seu respectivo domínio.

Vimos também que a distância Euclidiana precisou ser normalizada, pois cada atributo pertencia a um intervalo diferente. Logo, é preciso tomar o mesmo cuidado quando se pretende montar uma distância heterogenia, uma vez que os atributos não só podem estar em intervalos diferentes, mas também podem pertencer a domínios distintos.

Veremos agora HEOM, HVDM, DVDM, IVDM e NCM, algumas distâncias heterogenias. As quatro primeiras serão baseadas na distância euclidiana, distância de Hamming e VDM, já vistos. Já NCM, sobre o qual são propostas algumas alterações, é baseado num paradigma completamente diferente proposto por [Wang 06].

2.5 HEOM – Heterogeneous Euclidian-Overlap Metric

Essa distância combina a distância euclidiana para atributos numéricos e a distância de Hamming, também chamada *Overlap*, para atributos categóricos. A distância entre dois vetores de atributos é raiz quadrada da soma dos quadrados das distâncias entre cada um dos atributos. Para que cada um desses atributos não tenha uma influência maior, a distância entre dois atributos é normalizada. A distância entre dois atributos possui o valor mínimo igual a 0 e o valor máximo igual a 1. Como visto a distância de Hamming responde com 0 se os dois atributos possuem o mesmo valor e com 1 se possuem valores diferentes. Para usar a distância euclidiana, tem-se que fazer a normalização devida vista na seção 2.2, onde se divide a diferença entre dois atributos pela diferença máxima possível, assim o valor dessa razão será 1 se a diferença for a máxima permitida e 0 se os atributos possuem o mesmo valor.

No caso de atributos com valores desconhecidos dois casos precisam ser tratados: o primeiro é quando apenas uns dos atributos têm valor desconhecido, a distância entre eles é assumida como 1 (máxima possível); e segundo, se os dois atributos têm valores desconhecidos, a distância entre eles é 0. A partir dessa última consideração é que se torna possível que qualquer elemento comparado com ele mesmo possua uma distância igual a zero. Essa forma de tratar os dados com valores desconhecidos se assemelha muito à distância de Hamming e será utilizada também nas distâncias HVDM, DVDM e IVDM.

Podemos definir *HEOM(x,y)*, como uma função de distância entre dois elementos x e y com as equações abaixo:

$$HEOM(x,y) = \sqrt{\sum_{a=1}^{n} heom_a(x_a, y_a)^2}$$

onde,

$$heom_a(x_a, y_a) = \begin{cases} 1, \textit{ se } x_a \otimes y_a \textit{ é desconhecido} \\ 0, \textit{ se } x_a \wedge y_a \textit{ são desconhecidos} \\ h_a(x_a, y_a), \textit{ se a é categórico} \\ dif_a(x_a, y_a), \textit{ se a é numérico} \end{cases}$$

$$h_a(x_a, y_a) = \begin{cases} 1, \textit{ se } x_a \neq y_a \\ 0, \textit{ se } x_a = y_a \end{cases}$$

$$dif_a(x_a, y_a) = \frac{|x_a - y_a|}{\max(a) - \min(a)}$$

2.6 HVDM – Heterogeneous Value Difference Metric

Essa distância combina distância euclidiana para atributos numéricos e vdm_a para atributos categóricos. Difere de HEOM apenas pelo fato de utilizar vdm_a ao invés de Hamming para calcular a distância ente atributos categóricos.

Podemos definir *HVDM(x,y)*, como uma função de distância entre dois elementos x e y com as equações abaixo:

$$HVDM(x,y) = \sqrt{\sum_{a=1}^{n} hvdm_a(x_a, y_a)^2}$$

onde,

$$hvdm_a(x_a, y_a) = \begin{cases} 1, \textit{ se } x_a \otimes y_a \textit{ é desconhecido} \\ 0, \textit{ se } x_a \wedge y_a \textit{ são desconhecidos} \\ vdm_a(x_a, y_a), \textit{ se a é categórico} \\ dif_a(x_a, y_a), \textit{ se a é numérico} \end{cases}$$

$$vdm_a(x, y) = \sum_{c=1}^{C} \left| P_{a,x,c} - P_{a,y,c} \right|^q$$

$$dif_a(x_a, y_a) = \frac{\left| x_a - y_a \right|}{\max(a) - \min(a)}$$

2.7 DVDM – Discretized Value Difference Metric

DVDM faz uso de vdm_a tanto para atributos categóricos como para atributos numéricos. Contudo vdm_a não pode ser diretamente aplicado a atributos numéricos. Se tentássemos fazer isso encontraríamos casos onde $N_{a,x} = 1$, ou seja, somente uma instância iria ter aquele valor x para o atributo a, e $N_{a,x,c} = 0$ para toda classe c que não fosse a classe do elemento em questão. Logo não teria com quem comparar os atributos pela distribuição nas classes. Esse problema pode ser que não aconteça quando os atributos numéricos são discretos e estiverem distribuídos num intervalo relativamente estreito. E é certo que tal dificuldade aconteça se os valores numéricos forem contínuos.

Para resolver esse problema é proposta uma discretização, um processo que associe um rótulo a um valor numérico. De modo que os rótulos gerados apresentem representatividade plausível nas classes da base de dados. Se o número de rótulos criados é '*s*', para valores grandes de *s* não estamos resolvendo o problema que tínhamos, e para valores muito pequenos fica difícil de comparar a distribuição entre as classes. Uma alternativa para determinar o valor mais adequado para *s* é realizar testes com o conjunto de treinamento, porém isso pode ser um tanto custoso. A fórmula utilizada para discretização é dada abaixo, *s* é o número de rótulos gerados:

$$discretize_a(x_a) = \begin{cases} xa, \textit{ se a é categórico} \\ \left\lfloor s \times \dfrac{x_a - \min(a)}{|\max(a) - \min(a)|} \right\rfloor + 1, \textit{ se a é numérico} \end{cases}$$

Em testes realizados, foi escolhido nesse trabalho o valor de s = 10 por apresentar bons resultados. Vê-se da fórmula que os rótulos gerados serão os números inteiros de 1 a 10.

Agora podemos definir *DVDM(x,y)*, como uma função de distância entre dois elementos x e y com as equações abaixo:

$$DVDM(x, y) = \sqrt{\sum_{a=1}^{n} dvdm_a(x_a, y_a)^2}$$

onde,

$$dvdm_a(x_a, y_a) = \begin{cases} 1, \textit{ se } x_a \otimes y_a \textit{ é desconhecido} \\ 0, \textit{ se } x_a \wedge y_a \textit{ são desconhecidos} \\ vdm_a(discretize(x_a), discrestize(y_a)), \textit{ para os outros casos} \end{cases}$$

$$vdm_a(x, y) = \sum_{c=1}^{C} \left| P_{a,x,c} - P_{a,y,c} \right|^q$$

2.7 IVDM – Interpolated Value Difference Metric

IVDM também utiliza vdm_a para atributos categóricos e numéricos, difere de DVDM quando usado para atributos numéricos, pois calcula a probabilidade a priori utilizada no vdm_a interpolando a partir da probabilidade dos dois discretizados mais próximos.

Definimos *IVDM(x,y)*, como uma função de distância entre dois elementos x e y com as equações abaixo:

$$IVDM(x,y) = \sqrt{\sum_{a=1}^{n} ivdm_a(x_a, y_a)^2}$$

onde,

$$ivdm_a(x_a, y_a) = \begin{cases} 1, \textit{ se } x_a \otimes y_a \textit{ é desconhecido} \\ 0, \textit{ se } x_a \wedge y_a \textit{ são desconhecidos} \\ vdm_a(x_a, y_a), \textit{ se a é categórico} \\ \sum_{c=1}^{C} \left| P_{a,c}(x_a) - P_{a,c}(y_a) \right|^2, \textit{ se a é numérico} \end{cases}$$

$$vdm_a(x,y) = \sum_{c=1}^{C} \left| P_{a,x,c} - P_{a,y,c} \right|^q$$

$P_{a,c}(x_a)$ é a probabilidade a priori interpolada entre $P_{a,u,c}$ e $P_{a,u+1,c}$, onde u é x_a discretizado e $u+1$ é o próximo rótulo na seqüência dos rótulos discretizados.

$$P_{a,c}(x_a) = P_{a,u,c} + \left(\frac{x - meio_{a,u}}{meio_{a,u+1} - meio_{a,u}} \right) \times \left(P_{a,u+1,c} - P_{a,u,c} \right)$$

$$u = discretize(x_a)$$

$$meio_{a,u} = \min(a) + \left(\max(a) - \min(a) \right) \times \left(u + 0.5 \right)$$

2.8 NCM – Neighborhood Counting Measure

O NCM foi proposto por [Wang 06] e é na verdade uma medida de similaridade, baseada na contagem de vizinhanças. Vizinhanças são regiões do espaço, no caso do espaço onde estão os elementos da base de treinamento. NCM é uma medida que indica que dois elementos são mais semelhantes se eles têm mais vizinhanças em comum.

Para poder usar NCM como uma medida, é preciso primeiro definir um conceito de vizinhança. Vamos definir que uma vizinhança (uma região no espaço) como um conjunto de intervalos – um intervalo para cada atributo. Se o atributo for numérico um intervalo é fácil de definir. Se for um atributo categórico definimos um intervalo como um conjunto de possíveis valores. Na **Figura 1** vemos o que representa a distancia entre dois elementos e na **Figura 2** exemplo de vizinhanças, intervalos no espaço que contém pontos (elementos).

Figura 1. Distância entre elementos num espaço bidimensional.

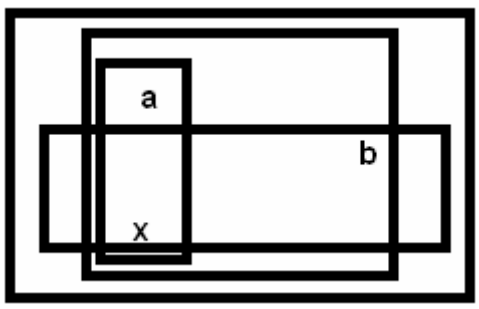

Figura 2. Vizinhanças num espaço bidimensional.

Um elemento pertencer a uma vizinhança se para cada atributo a o valor do seu atributo está no intervalo equivalente àquele atributo. Um valor numérico x_a pertence a um intervalo $[x_{a1},x_{a2}]$ se $x_{a1} \leq x_a \leq x_{a2}$. Um valor categórico pertence a um intervalo se ele é um dos elementos daquele conjunto.

Dois elementos são mais semelhantes quanto mais vizinhas em comum eles têm. Uma maneira de calcular quantas vizinhanças dois elementos têm em comum é listar todas vizinhanças possíveis e ver quais delas contém ambos os elementos. Contudo essa é uma tarefa muito custosa. Uma maneira mais fácil é fazer a contagem de vizinhanças em comum através de análise combinatória.

O número de vizinhanças possíveis para um valor de um atributo categórico é igual ao número de conjuntos possíveis que contém aquele valor específico para o atributo, isto é, 2^{m_a-1}, onde $m_a = |dom(a)|$. Relembrando que *dom(a)* é o conjunto que contém todos os possíveis valores para o atributo categórico da posição *a*, e $|dom(a)|$ é a cardinalidade desse conjunto, o número de elementos que ele possui. Também é possível contar quantas vizinhas em comum existem para dois atributos categóricos, se esse dois atributos são iguais o número de vizinhas em comum é igual ao número de vizinhanças possíveis para um deles, isto é, 2^{m_a-1}. Se esses atributos forem diferentes o número de possíveis conjuntos contendo os dois valores é 2^{m_a-2}.

A quantidade de vizinhanças para um valor de um atributo numérico é a contagem dos possíveis intervalos que contêm aquele valor. Descobrimos esse valor como o produto dos possíveis limitantes inferiores e superiores a esse número, sem nos esquecer que por ser atributo numérico ele assume um valor máximo e mínimo – max(*a*) e min(*a*) – na base de dados. Não é possível definir todos os limitantes superiores e inferiores para um número contínuo, contudo vamos definir como calcular esse resultado para números discretos e utilizar a mesma fórmula para ambos os casos. O número de vizinhanças de um atributo numérico é $(\max(a) - x_a + 1) \times (x_a - \min(a) + 1)$. Igualmente podemos contar qual o número de intervalos que contêm dois valores, o que equivale ao número de vizinhanças em comum a dois pontos. Para fazer isso calculamos o número de limitantes inferiores do menor dos dois números e multiplicamos pelo

número de limitantes superiores do maior deles, esse valor é igual a $(\max(a) - \max(\{x_a, y_a\}) + 1) \times (\min(\{x_a, y_a\}) - \min(a) + 1)$.

Uma vez que conseguimos calcular o número de vizinhanças em comum a dois atributos quaisquer podemos definir $viz_a(x_a,y_a)$ como sendo a função que conta as vizinhanças comuns a dois atributos.

$$viz_a(x_a, y_a) = \begin{cases} 2^{m_a-1}, \textit{ se a é categórico e } x_a = y_a \\ 2^{m_a-2}, \textit{ se a é categórico e } x_a \neq y_a \\ (\max(a) - \max(\{x_a, y_a\}) + 1) \times (\min(\{x_a, y_a\}) - \min(a) + 1), \textit{ se a é numérico} \end{cases}$$

O número de vizinhanças comuns a dois vetores de atributos, conforme a análise combinatória é o produto de todas as possíveis vizinhas de cada atributo, ou seja, o produto de $viz_a(x_a,y_a)$ para todo atributo a, que chamamos de $NCM'(x,y)$.

$$NCM'(x, y) = \prod_{a=1}^{n} viz_a(x_a, y_a)$$

Porém, esse número é muito grande e muito dependente de cada atributo, aqui também é preciso fazer uma normalização. Para tanto vamos definir $viz_a(x_a) \geq viz_a(x_a,y_a)$ o número de vizinhanças que contém o valor x_a. Então $viz_a(x_a,y_a)/viz_a(x_a)$ será um valor entre 0 e 1. Um $NCM'(x,y)$ normalizado chamamos $NCM(x,y)$ é definido como:

$$NCM(x, y) = \prod_{a=1}^{n} \frac{viz_a(x_a, y_a)}{viz_a(x_a)}$$

Contudo essa é uma medida de similaridade, que é simetricamente oposta à distância. Em $NCM(x,y)$ dois elementos estão o mais próximo o possível valor obtido é o máximo possível 1, se fosse uma distância seria 0. Também para dois elementos o mais distante possível $NCM(x,y)$ responde com 0 e se fosse uma distância responderia com 1. Para transformarmos essa media em distância criamos $NCM1(x,y)$.

$$NCM1(x, y) = 1 - NCM(x, y)$$

$$NCM1(x,y) = 1 - \prod_{a=1}^{n} \frac{viz_a(x_a, y_a)}{viz_a(x_a)}$$

Percebe-se que essa distância não é simétrica, ou seja, $NCM1(x,y) \neq NCM1(y,x)$. Pois a normalização é feita com base apenas no número de vizinhos de um dos elementos. Segundo [Theodoridis; Koutroumbas 03] se uma medida de distância não é simétrica ela não pode se considerada uma métrica. Para tanto quatro propostas são aqui apresentadas para criar medidas de distância simétricas baseado na contagem de vizinhanças.

A primeira proposta é $NCM2(x,y)$ onde a normalização de $viz_a(x_a,y_a)$ é feita pela média aritmética de $viz_a(x_a)$ e $viz_a(y_a)$.

$$NCM2(x,y) = 1 - \prod_{a=1}^{n} \frac{viz_a(x_a, y_a)}{\left(viz_a(x_a) + viz_a(y_a)\right)/2}$$

A segunda proposta é $NCM3(x,y)$ onde a normalização de $viz_a(x_a,y_a)$ é feita pela média geométrica de $viz_a(x_a)$ e $viz_a(y_a)$.

$$NCM3(x,y) = 1 - \prod_{a=1}^{n} \frac{viz_a(x_a, y_a)}{\sqrt{viz_a(x_a) viz_a(y_a)}}$$

Em $NCMm(x,y)$ onde a normalização de $viz_a(x_a,y_a)$ é feita pelo menor valor entre $viz_a(x_a)$ e $viz_a(y_a)$.

$$NCMm(x,y) = 1 - \prod_{a=1}^{n} \frac{viz_a(x_a, y_a)}{\min(viz_a(x_a), viz_a(y_a))}$$

E em $NCMM(x,y)$ onde a normalização de $viz_a(x_a,y_a)$ é feita pelo maior valor entre $viz_a(x_a)$ e $viz_a(y_a)$.

$$NCMM(x,y) = 1 - \prod_{a=1}^{n} \frac{viz_a(x_a, y_a)}{\max(viz_a(x_a), viz_a(y_a))}$$

3. Algoritmos

Nesta seção veremos os dois algoritmos de classificação onde serão empregas as funções de distância do capítulo 2: o k-NN e a rede neural RBF. Não se pretende aprofundar em características desses algoritmos, somente explicar o seu funcionamento, mostrar os pontos onde se faz o uso do cálculo de distância e explicitar os detalhes de implemtação.

3.1 k-NN

O k-NN é um dos algoritmos de classificação mais simples. De fácil entendimento e implementação. Sua idéia consiste em calcular a distância de uma dada instância, da qual pretende-se descobrir a classe, para todos os elementos da base de treinamento. Conhecendo essas distâncias escolher os k-vizinhos mais próximos (*k-Nearet Neighbor, k-NN*), que são os k elementos que possuem a menor distância para a instância em questão. Sabendo quem são os k-NN aplica-se uma regra de classificação para inferir a classe.

Na literatura duas regras são muito comuns: a maioria por votação e o peso pela distância. Ainda propomos aqui uma regra de classificação chamada perda de energia, que também considera a distância para calcular o peso.

Maioria na Votação (sem peso)

Essa regra de classificação atribui à instância em questão a classe que estiver em maior quantidade entre os k-NN. Na **Figura 3** vemos um 3-NN tentando classificar o padrão *x*, pela regra de maioria de votação ele pertence à classe a. Já o 7-NN **Figura 4** atribui x à classe b.

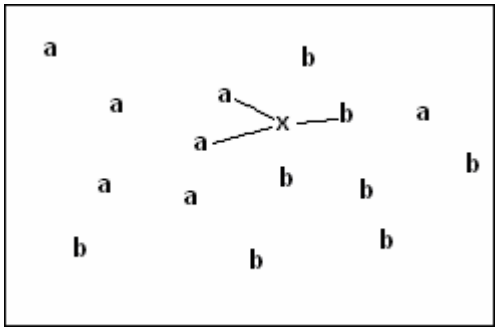

Figura 3. Exemplo gráfico de 3-NN num espaço bidimensional

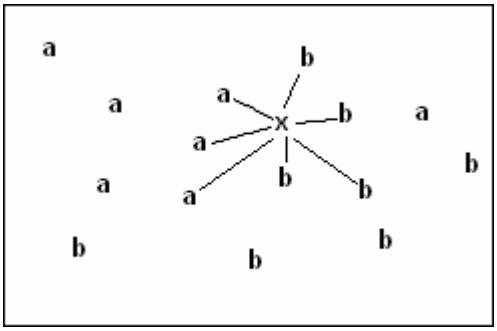

Figura 4. Exemplo gráfico de 7-NN num espaço bidimensional

Peso pela Distância

Vê-se que a regra de classificação sem peso (maioria na votação) não se demonstra muito robusta, pois só considera as classes do k-NN sem levar em conta que alguns elementos estão mais próximos do que outros da instância investigada. A regra de peso pela distância atribui a cada um dos k-NN um peso proporcional a sua distância, sendo que o peso do elemento mais próximo é $w_1 = 1$ e do elemento mais distante é $w_{\underline{k}} = 0$. Os pesos do outros k-NN de um padrão t podem ser calculados pela fórmula:

$$w_i = \frac{d(x_k,t) - d(x_i,t)}{d(x_k,t) - d(x_1,t)}$$

Onde x_k é o elemento mais distante de t dentre os k-NN e x_1 é o elemento mais próximo; x_i é algum dos k-NN de t; e $d(x,t)$ é a distância entre x e t.

Para cada classe é atribuída a soma de pesos w_i para todo x_i que pertence àquela classe. A classe que obtiver a maior soma de pesos será a classe atribuída ao padrão t.

Perda de Energia (energia)

Esse regra foi inspirada no fenômeno natural de propagação de ondas mecânicas, como ondas sonoras. A energia de uma onda dessas decai na razão de $1/d^2$, onde d é a distância até a fonte, de onde a onda foi emitida. Esse decaimento representa a energia que antes estava concentrada numa pequena região e agora está dispersa num espaço maior. É como se a energia antes estivesse concentrada na casca de uma pequena esfera e com o aumento do raio dessa esfera a energia se espalha pela casca que se dilata com o aumento do raio (distância ao centro ou à fonte). A casca da esfera dilata na razão de d^2, por isso a energia diminui com $1/d^2$.

Supomos agora que esse fenômeno ocorra num espaço n-dimensional e que a energia se dissipe na casca de uma hiper-esfera, essa perda de energia acontece na razão de $1/d^{n-1}$.

O funcionamento da regra de classificação por energia (perda de energia) funciona como a classificação por peso, ou seja, atribui para cada classe o peso dos k-NN que pertencer àquela classe. Sendo que o peso w_i agora é calculado da seguinte forma:

$$w_i = \frac{1}{d(x_i,t)^n}$$

Onde x_i é algum dos k-NN do padrão t que se pretende classificar; e $d(x,t)$ é a distância entre x e t. Foi usado $1/d^n$ ao invés de $1/d^{n-1}$ por questão de simplificação. Uma vez que o efeito principal dessa forma de se calcular os pesos é dar uma importância muito maior para os elementos mais próximos e bem menor aos mais distantes. É isso que faz a operação de potenciação.

3.2 RBF treinada com DDA

As Redes Neurais de Função de Base Radial (RBF – Radial Basis Function) são bastante utilizadas por necessitarem de poucas épocas para treinamento. Estas redes, assim como diversos algoritmos de aprendizagem de máquina e também redes de outros paradigmas, necessitam de uma medida de similaridade/dissimilaridade para os padrões apresentados.

O que essa rede faz na prática é dividir o espaço dos elementos da base de treinamento em regiões para cada classe. Essas regiões são definidas como um conjunto de hiper-esferas. Para tal escolhem-se os centros e os respectivos raios dessas hiper-esferas. As funções de base radial vão fazer com que a resposta para a pertinência a cada uma dessas regiões seja mais suave, isto é, não responde somente com 0 ou 1, mas com um valor proporcional à distância ao centro. No caso da função Gaussiana, utilizada aqui, apresenta as maiores respostas quando a distância é bem próxima de 0 e respostas cada vez menores quando a distância ao centro vai aumentando (**Figura 5)**.

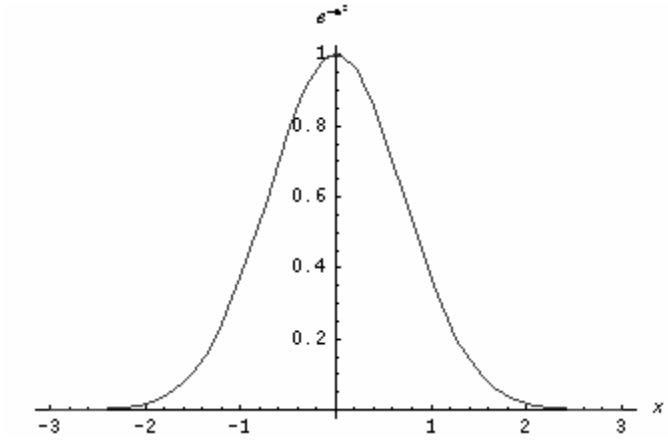

Figura 5. Gráfico da Função Gaussiana. Ela apresenta os maiores valores perto do 0 e valores cada vez menores quando vai se afastando do 0. A resposta máxima para essa função é 1 e responde com valores bem próximo de 0 quando para um entrada maior que o parâmetro raio.

Essas redes possuem duas camadas: a camada intermediária e a camada de saída. As funções de base radial são utilizadas na camada intermediária. Os parâmetros que precisam ser treinados nessa rede são três: os centros e os raios das funções de base radial e os pesos entre os neurônios intermediários e os da saída. **Figura 6**.

Erro!

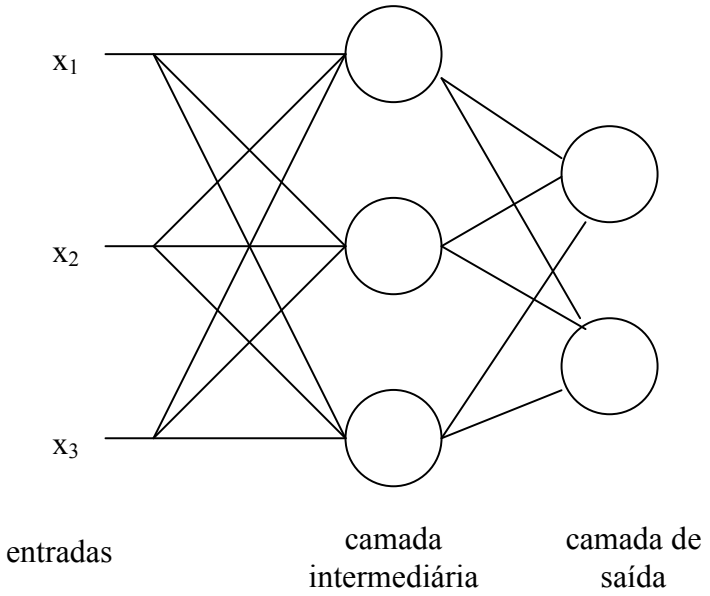

Figura 6. Arquitetura de uma rede RBF.

Os neurônios da camada intermediária utilizando a função de ativação Gaussiana respondem para um padrão *t* com $R_i(t)$:

$$R_i(t) = \exp\left(-\frac{d(x_i,t)^2}{r_i^2} \right)$$

Onde $R_i(t)$ é a resposta, no intervalo [0,1] do i-ésimo neurônio da camada intermediária; x_i é o centro e r_i é o raio da hiper-esfera representada por esse neurônio; e $d(x_i,t)$ é a distância entre *x* e *t*.

Os neurônios da camada de saída respondem com $S_c(t)$:

$$S_c(t) = \frac{\sum_{i=1}^{m_c} A_i^c R_i(t)}{\sum_{i=1}^{m_c} A_i^c}$$

Onde $S_c(t)$ é a resposta do neurônio da camada de saída que identifica a classe c; A_i^c é o peso entre o neurônio i da camada intermediária e o neurônio c da camada de saída; m_c é o número de neurônios da camada intermediária que se ligam com o neurônio c da camada de saída. Os neurônios da camada intermediária só se ligam a um neurônio da camada de saída, uma vez que cada um deles delimita uma região no espaço para uma classe, não faz sentido que se liguem com mais de um neurônio da camada de saída. A resposta de $S_c(t)$ é um valor no intervalo [0,1], uma vez que é a média ponderada das respostas dos neurônios da cada intermediária, que estão no mesmo intervalo.

Uma alternativa a essa função de ativação na camada de saída é aplicar a esse resultado a função Sigmóide Logística, desse modo podemos definir $S'_c(t)$ como sendo a função Sigmóide Logística aplicada a $S_c(t)$:

$$S'_c(t) = Sigmóide\left(S_c(t)\right)$$

$$Sigmóide(x) = \frac{1}{1 + \exp\left((x - 0.5) \times (1 - z)\right)}$$

Treinamento

Para o treinamento da camada intermediária comumente utilizam-se métodos não-supervisionados de aprendizagem, como algoritmos de agrupamento e para a camada de saída, aprendizagem supervisionada, como métodos de correção de erros.

Neste trabalho decidimos utilizar um esquema construtivo para que as comparações com diferentes bases pudessem ser feitas utilizando os mesmos parâmetros. Desta forma a rede foi treinada utilizando o algoritmo DDA, exibido na seção seguinte.

DDA - Dynamic Decay Adjustment

Esse método foi proposto por [Berthold; Diamond 95] e treina todos os parâmetro da rede: os centros e os raios das funções de base radial e os pesos entre os neurônios intermediários e os da saída.

Para realizar o treinamento não faz uso de taxa de aprendizagem, mas possui dois parâmetros θ^+ e θ^-. Esses parâmetros servem para definir os tamanhos dos raios das funções de base radial. Pois para um dado padrão *t* pertencente à classe *c*. A resposta de algum neurônio da camada intermediária da classe *c* deve ser maior que θ^+. E a resposta de cada neurônio da classe $d \neq c$ ao padrão *t* deve ser menor que θ^-. Assim as duas condições que precisam ser garantidas são:

$$\exists i : R_i^c(t) \geq \theta^+$$

$$\forall d \neq c, 1 \leq j \leq m_k : R_i^c(t) < \theta^+$$

Para construir uma rede neural RBF que satisfaz a seguinte condição pode-se usar o algoritmo:

FORALL padrão de treino (t,c) **DO**:
IF $\exists N_i^c : R_i^c(t) \geq \theta^+$ **THEN**
$A_i^c += 1.0$
ELSE
Crie um novo neurônio $N_{m_{c+1}}^c$ com:

- o centro do novo neurônio é $x_{m_{c+1}}^c = t$
- o raio dele é o máximo possível satisfazendo a segunda restrição
  $r_{m_{c+1}}^c = \max_{d \neq c, 1 \leq j \leq m_d} \{r : R_{m_{c+1}}^c(x_j^d) < \theta^-\}$

- o peso inicial entre esse neurônio de e a camada de saída é $A^c_{m_{c+1}} = 1.0$
- $m_c+ = 1$

ENDIF

FORALL $\forall d \neq c, 1 \leq j \leq m_k$ DO

//Ajuste os raios dos neurônios das outras classes de modo que t tenha uma
//resposta menor que θ^- para todos eles.

$r^d_{j_1} = \max\{r : R^d_j(t) < \theta^-\}$

ENDFOR

Os padrões de treinamento são apresentados à rede um a um. Se o novo padrão apresentado é classificado corretamente, o peso do entre o neurônio que apresentou a melhor resposta e a camada de saída é aumentado. Se o padrão não é classificado corretamente um novo neurônio é criado, onde o centro é o padrão que está sendo apresentado na iteração atual e o raio é o maior possível que atende às restrições estabelecidas. O último passo do algoritmo é garantir a segunda restrição para os neurônios que já estavam na rede.

A escolhas dos valores de θ^+ e θ^- geralmente não é muito critica segundo [Berthold; Diamond 95], mas [Oliveira *et al* 05] diz que é possível obter melhores resultados se bem escolhido o valor de θ^-. Os valores utilizados nesse trabalho foram $\theta^+ = 0.4$ e $\theta^- = 0.1$.

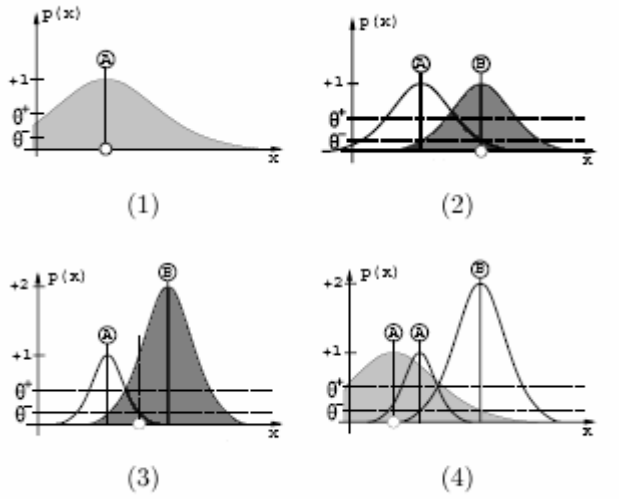

Figura 7. Um exemplo do algoritmo DDA: (1) um padrão da classe A é apresentado à rede e um novo neurônio é criado; (2) depois é apresentado um padrão da classe B e também se cria um novo neurônio para representá-lo, em seguida o raio do neurônio da classe A é ajustado para não conflitar com o padrão da classe B; (3) um outro padrão da classe B é classificado corretamente, aumentando o peso do neurônio que classificou esse padrão e ainda o raio de do neurônio da classe A é classificado corretamente; (4) um novo padrão da classe A não é reconhecido pelo neurônio da mesma classe na rede e é necessário criar um novo neurônio para reconhecê-lo.

4. Resultados

Para testar os algoritmos descritos acima empregando as funções de distâncias pesquisadas, foram utilizadas 14 bases do UCI *Machine Learning Repository* [UCI 07]. O método de teste foi o *10-fold cross validation* descrito em [Kohavi 95].

O *10-fold cross validation* consiste em dividir a base de dados em 10 partes de aproximadamente o mesmo tamanho, daí realizar 10 testes, usando em cada teste uma parte como conjunto de testes e as outras 9 como conjunto de treinamento.

Para cada taxa de acerto computada neste trabalho foram realizados 10 testes do tipo *10-fold cross validation*, o que dá um montante de 100 testes sobre os quais foram realizados os cálculos da média de acerto e o respectivo desvio padrão – no caso do RBF, como veremos, foram realizadas outras medidas além destas. Vale salientar que antes de cada *10-fold cross validation* a base é "embaralhada" aleatoriamente.

As 14 bases retiradas do UCI Machine Learning Repository[], como visto na são bem distintas umas das outras. Existem bases cujas instâncias possuem somente atributos categóricos, somente numéricos, ou ambos. Bases com poucos e muitos atributos. Poucas e muitas classes. Bases com e sem *missing data* – atributos com valor desconhecidos.

4.1 Testes com o k-NN

Os testes com o k-NN empregando as funções de distâncias listadas é dividido em duas partes. Primeiro são mostrados os gráfico dos resultados obtidos na média das 14 bases utilizadas. Em seguida algumas bases dão destacadas por apresentar comportamento bem distinto da média.

Base	Elemts.	Classes	Atrib.	Num.	Categ.	Desc.
Audiology (AudS)	200	24	69	0	69	Sim
Primary Tumor (PrimT)	339	22	17	0	17	Sim
Zoo	101	7	16	0	16	Não
3 bases de atributos categóricos						
Ecoli	336	8	7	7	0	Não
Glass	214	7	9	9	0	Não
Ionosphere (Iono)	351	2	34	34	0	Não
Iris	150	3	4	4	0	Não
Sonar	208	2	60	60	0	Não
Wine	178	3	13	13	0	Não
6 bases de atributos numéricos						
Auto	205	6	24	15	9	Sim
Breast-Cancer (BrsC)	286	2	9	1	8	Sim
Heart-Statlog (Heart)	270	2	13	6	7	Não
Hepatitis(Hepatt)	155	2	19	6	13	Sim
Horse-Colic (HorseC)	300	2	26	7	19	Sim
5 bases de atributos mistos						
14 bases de dados						

Tabela 2. Tabela das bases de dados utilizadas nos testes. O primeiro campo contém o nome da base e uma abreviação desse nome entre parênteses; Elemts. Diz o número de instâncias de cada base; Classes o número total de classes; Atrib. O número total de atributos; Num. e Categ. a quantidade de atributo numéricos e categóricos; Desc. Indica se tem ou não *missing data.*

Os valores de k empregados fora: 1, 6, 11, 16, 21, 31 e max (max = todos os elementos do conjunto de treinamento). Também foram observadas as três regras de classificação descritas: sem peso, com peso e energia (por perda de energia).

O k-NN **sem peso** mostrado na **Tabela 3** apresenta na média seus melhores resultados para o valor de k = 1. Os resultados são semelhantes em todas as funções de distância, o que reforça a teoria de que não existe a melhor função de distância e sim uma que se adeqüe melhor a uma base específica. Observamos ainda, como era de se esperar, a mesma taxa de acerto (44%) quando k = max. Uma vez que a regra de classificação sem peso atribui a uma dada instância a classe de maior freqüência entre os k vizinhos, quando os k vizinhos são o conjunto de treinamento inteiro (k = max) a classificação vai ser sempre o elemento que a é moda desse conjunto e isso independe de função de distância.

KNN - [sem peso]							
Média	k = 1	k = 6	k = 11	k = 16	k = 21	k = 31	k = max
HEOM	76,9	76,1	74,8	73,6	73,3	72,0	44,0
HVDM	77,9	76,7	75,4	74,4	73,5	71,7	44,0
DVDM	77,2	76,8	76,0	74,6	73,5	72,4	44,0
IVDM	78,0	77,0	76,2	75,1	73,8	72,7	44,0
NCM1	75,8	75,4	74,5	73,8	73,5	72,6	44,0
NCM2	76,7	76,0	75,2	74,6	74,2	73,0	44,0
NCM3	76,8	76,1	75,2	74,6	74,3	73,1	44,0
NCMm	76,5	75,8	75,6	74,5	73,9	73,0	44,0
NCMM	76,5	76,3	75,3	74,7	74,3	72,8	44,0

Tabela 3. k-NN com a regra de classificação sem peso

A média do k-NN **com peso** mostradas na **Tabela 4** obteve a maioria dos seus resultados entre 75 e 80%, diferentemente do k-NN sem peso que está entre 70 e 75%, o que mostra que essa regra de classificação é mais robusta. Ainda, ao contrário da regra sem peso, o melhores resultados foram obtidos com k > 1. As funções HEOM, HVDM, DVDM e IVDM apresentaram seu máximo para k = 11 e uma taxa de acerto muito inferior aos outros valores de k quando k = max – mais de 15 pontos de diferença. Enquanto as variações de NCM obtiveram melhores resultados quando k = 21 e pouca diferença quando k = max – cerca de 4 pontos de diferença.

Podemos também analisar a sensibilidade a dados ruidosos quando comparamos a variação de acerto máxima com o acerto para k = max. E NCM mostrou-se menos sensível a dados espúrios quando usado com essa regra de classificação.

KNN - [com peso]							
Média	k = 1	k = 6	**k = 11**	k = 16	**k = 21**	k = 31	**k = max**
HEOM	76,9	78,4	**78,4**	78,4	78,1	77,2	**61,5**
HVDM	77,9	79,2	**79,5**	79,2	78,7	77,6	**61,8**
DVDM	77,2	78,3	**78,9**	78,7	78,3	77,7	**65,7**
IVDM	78,0	79,3	**79,5**	79,0	78,7	77,8	**64,8**
NCM1	75,8	77,9	78,4	78,7	**78,9**	78,7	**74,1**
NCM2	76,7	78,4	78,9	79,1	**79,2**	79,2	**75,0**
NCM3	76,8	78,5	78,9	79,2	**79,3**	79,2	**75,1**
NCMm	76,5	78,4	79,0	79,3	**79,6**	79,5	**74,7**
NCMM	76,5	78,3	78,8	79,1	**79,2**	79,2	**75,2**

Tabela 4. k-NN com a regra de classificação com peso

Nos experimentos usando a regra de classificação de **energia** (perda de energia) vê-se claramente que tal regra mostra-se bastante insensível a dados ruidosos. Para as funções HVDM e IVDM apresentou seu ótimo para k = max. Em HEOM e DVDM o máximo de acerto ficou em k = 31, porém a diferença do acerto é desprezível. NCM e variações conseguiram seu máximo em k = 11, também com pouca variação – até 4 pontos – entre esses valores e o acerto com k = max.

Conforme na regra com peso as funções NCM mostraram-se semelhante sensibilidade a ruído quando combinadas com a regra de energia, contudo aparentam maior variação se comparada às outras funções de distância.

Pode-se concluir desses três resultados que diferentes funções de distância apresentam alcançam resultados também distintos, abrindo oportunidades para se empregar a função que se apresentar melhor, conforme os critérios de "melhor" estabelecidos.

Analisando os resultados em três regras de classificação infere-se também que uma mesma função de distância apresenta comportamentos diferentes de algoritmo para algoritmo.

KNN - [energia]							
Média	k = 1	k = 6	**k = 11**	k = 16	k = 21	**k = 31**	**k = max**
HEOM	76,9	77,8	77,8	77,9	77,9	**78,0**	**77,8**
HVDM	77,9	78,2	78,2	78,2	78,2	**78,2**	**78,2**
DVDM	77,2	77,6	77,7	77,6	77,6	**77,7**	**77,5**
IVDM	78,0	78,4	78,6	78,6	78,6	**78,6**	**78,7**
NCM1	75,8	76,8	**76,8**	76,8	76,7	76,7	**74,7**
NCM2	76,7	77,4	**77,6**	77,5	77,4	77,3	**74,6**
NCM3	76,8	77,5	**77,7**	77,5	77,4	77,3	**74,7**
NCMm	76,5	77,3	**77,3**	77,3	77,1	77,2	**75,4**
NCMM	76,5	77,4	**77,4**	77,4	77,2	77,0	**73,3**

Tabela 5. k-NN com a regra de classificação energia

As médias dos acertos nas 14 bases são úteis para ser ter uma idéia do comportamento das funções de distância e das regras de classificação. Contudo, como já foi dito, cada base se comporta de uma forma particular – às vezes bastante distinta – mediante as mesmas situações sob as quais foram analisadas as taxas de acerto média. Por isso é bastante útil analisar algumas bases separadamente e observar os resultados possíveis.

Ao analisar as tabelas das medidas em dois pontos podem servir como referência: k = 1 e k = max (sem peso). Para k = 1 a taxa de acerto é a mesma independente da regra, uma vez que só considera o vizinho mais próximo, que só varia com a mudança da função de distância. Quando k = max e a regra de classificação é sem peso, ou seja, a cada elemento de teste é atribuída a classe de maior freqüência no conjunto de treinamento. Se a distribuição das classes na base for muito desbalanceada – contendo muita mais instancia de uma classe do que de outras – essa distribuição interfere na classificação do k-NN.

Como exemplo pode-se citar a base **Breast-Cancer**, que possui duas classes sendo que uma delas corresponde a 70,3% das instâncias. A taxa de acerto para k = 1 é em torno de 67% e a taxa de acerto para k = max (sem peso) é 70,3%, o que coincide com a distribuição desta base. Para k = 1 percebe-se que a taxa de acerto é menor do que o esperado, menor mesmo do que responder sempre a maior freqüência, isso porque o grande desbalanceamento dessa base acaba influecinando negativamente o k-NN.

A base **Iris** possui três classes igualmente distribuídas (33,3%), a taxa de acerto para k = 1 é em torno de 95% e para k = max (sem peso) é cerca de 22%. Era comum de se esperar uma taxa de acerto de 33% nesse segundo caso, mas essa variação deve ser devida ao método de teste o *10-fold cross validation,* junto com a ordenação aleatória da base.

Existem três bases cujos vetores de atributos são puramente categóricos: Audiology, Primary Tumor e Zôo. Para essas três bases podemos resumir 3, as 9 funções de distância listadas: distancia de Hamming em HEOM; VDM em HVDM, DVDM e IVDM; e NCM que não apresenta variações para atributos categóricos.

Audiology apresenta maior taxa de acerto quanto utiliza VDM como função de distância. É importante ressaltar a degradação a resposta no esquema sem peso para valores de k maiores que um e a pouquíssima degradação quando se usa a regra de energia. HEOM e NCM mesmo melhorando sua taxa de acerto com peso e energia para outros valores de k, como k = 6, não batem o acerto de VDM.

Audiology [sem peso]	k = 1	k = 6	k = max
HEOM	**72,8**	61,7	19,9
HVDM	**77,5**	61,5	19,9
NCM1	**72,2**	62,5	19,9
Audiology [com peso]	k = 1	k = 6	k = max
HEOM	72,8	**75,7**	42,2
HVDM	**77,5**	76,8	44,9
NCM1	72,2	**73,7**	66,0
Audiology [energia]	k = 1	k = 6	k = max
HEOM	72,8	**75,3**	**75,1**
HVDM*	**77,5**	77,4	**77,4**
NCM1	72,2	**74,8**	73,8

Tabela 6. Taxas de acerto para a base Audiology em três regras de classificação diferentes.

Em **Zoo** também é possível observar comportamento semelhante ao de Audiology por apresentar melhores resultados com VDM e pela degradação no acerto em cada esquema de peso. Nesses dois casos vê-se que o esquema de energia apresentou-se mais tolerante a dados ruidosos do que as outras regras de classificação.

Zoo [sem peso]	k = 1	k = 21	k = 31	k = max
HEOM	**95,7**	83,1	76,7	40,6
HVDM	**96,8**	78,2	75,6	40,6
NCM1	**95,7**	83,1	76,7	40,6
Zoo [com peso]	k = 1	k = 21	k = 31	k = max
HEOM	**95,7**	94,5	92,7	76,7
HVDM	**96,8**	94,5	92,5	83,8
NCM1	**95,7**	94,7	95,1	94,4
Zoo [energia]	k = 1	k = 21	k = 31	k = max
HEOM	**95,7**	95,4	95,4	**95,7**
HVDM*	**96,8**	**96,8**	**96,8**	**96,8**
NCM1	95,7	**96,2**	**96,2**	95,7

Tabela 7. Taxas de acerto para a base Zoo em três regras de classificação diferentes.

O melhor de todos os resultados para a base **Primary Tumor (PrimT)** foi 46,9% para HEOM (k = 21 sem peso). O esquema de energia apresentou resultados muito ruins nessa base. Poderia pensar-se que isso se deve ao número de atributos que influencia no cálculo da energia, contudo essa base possui 17 atributos e a base Zôo possui um valor semelhante, 16, e apresenta efeito bem distinto.

PrimT [sem peso]	k = 1	k = 6	k = 21	k = 31	k = max
HEOM*	35,5	43,2	**46,9**	45,3	24,8
HVDM	36,1	**42,7**	41,1	38,3	24,8
NCM1	33,7	42,2	**44,3**	43,6	24,8
PrimT [com peso]	k = 1	k = 6	k = 21	k = 31	k = max
HEOM	35,5	39,7	44,1	**45,0**	25,0
HVDM	36,1	38,9	43,9	**44,6**	24,8
NCM1	33,7	39,9	42,5	43,5	**45,2**
PrimT [energia]	k = 1	k = 6	k = 21	k = 31	k = max
HEOM	35,5	37,2	38,2	38,3	**38,4**
HVDM	**36,1**	34,4	34,4	34,5	34,6
NCM1	33,7	35,0	35,1	**35,3**	35,0

Tabela 8. Taxas de acerto para a base Primary Tumor em três regras de classificação diferentes.

A base **Auto** possui em cada instância da sua base 15 atributos numérico e 9 categóricos. Nas regras classificação com peso e sem peso sua taxa de acerto diminui para k > 1. Contundo apresenta sempre os melhores resultados para k = 16 com a regra de energia. IVDM e DVDM foram as funções de distancias que mais se adequaram a essa base.

Auto	k = 1	k=6[sp]	k=16[sp]	k=6[cp]	k=16[cp]	k=6[en]	k=16[en]
HEOM	**74,9**	61,8	55,2	73,5	66,3	**75,0**	**74,8**
HVDM	**77,4**	66,2	59,4	76,0	70,9	**77,2**	**77,2**
DVDM*	**79,8**	69,7	62,5	**79,0**	74,5	**79,9**	**80,0**
IVDM*	**80,4**	67,9	62,6	78,8	73,3	**80,8**	**81,0**
NCM1	**76,5**	62,9	55,1	75,8	75,8	**76,9**	**76,9**
NCM2	76,7	61,3	57,7	76,0	75,9	75,8	**77,8**
NCM3	76,8	60,8	57,1	76,4	75,9	76,4	**78,0**
NCMm	**77,4**	59,3	55,9	76,3	75,6	**78,2**	**77,6**
NCMM	**75,9**	62,6	58,2	74,9	75,2	74,0	**76,2**

Tabela 9. Taxas de acerto para a base Auto. [sp] = sem peso; [cp] = com peso; [en] = energia.

Heart-Statlog é uma base cujos elementos possuem 6 atributos numéricos e 7 categóricos. Nela torna-se visível que o valor de k também depende da função de distância, ratificando que isso é uma propriedade da base e poder ser semelhante em outros casos ou não. Observando apenas o esquema com peso as funções HVDM, DVDM e IVDM apresentam os resultados ótimos para a base quando o valor de k = 11, já NCM1, NCM2, NCM3 e NCMM quando k = max. HEOM e NCMm obtiveram resultados piores que os outros.

Esse simples gráfico (**Figura 8**) trás informações bem úteis. Se compararmos HEOM com HVDM, cuja única diferença é que o primeiro utiliza distância de Hamming para atributos categóricos e a outra utiliza VDM, vemos que o melhor desempenho de HVDM é devido à forma como ele trata os tais atributos. Pelo mesmo motivo o explicasse o sucesso de DVDM e IVDM, pois estes também usam VDM.

NCMm apresentou uma taxa de acerto de 82% outros NCM cerca de 84%, sendo que a diferença nesses algoritmos é na forma como tratam os atributos numéricos. Logo, o sucesso dos NCM é devido aos atributos numéricos. E por esses dados apresentarem um comportamento diferente dos categóricos, justifica-se também que os melhores resultados apareçam também com valor de k distinto do melhor para o VDM.

Aparentemente obter-se-ia melhores resultado se fosse possível combinar o VDM para atributos categóricos e o NCM2 para atributos numéricos. Contudo essas formas de cálculos de distância baseiam-se em dois paradigmas bastante distintos e a sua junção aparenta ser não trivial. Contudo, ainda que resolvido esse problema, lembramos que uma apresenta seu melhor para k = 11 e o outro para k = max. Então uma sugestão de melhoramento tão obvia fica escondida sob mecanismos matemáticos.

Muitas outras bases poderiam ser analisadas. Aqui estão presentes somente aquelas que enriquecem o objetivo do trabalho: mostrar como o cálculo da distância pode interferir na taxa de acerto de um classificador. O desvio padrão da taxa de acerto também sofre variações a depender da função de distância. Geralmente as maiores taxas de acerto apresentam o menor desvio padrão, contudo essa análise não é feita nesse documento. Essa e outras informações podem ser encontradas nas tabelas completas em anexo.

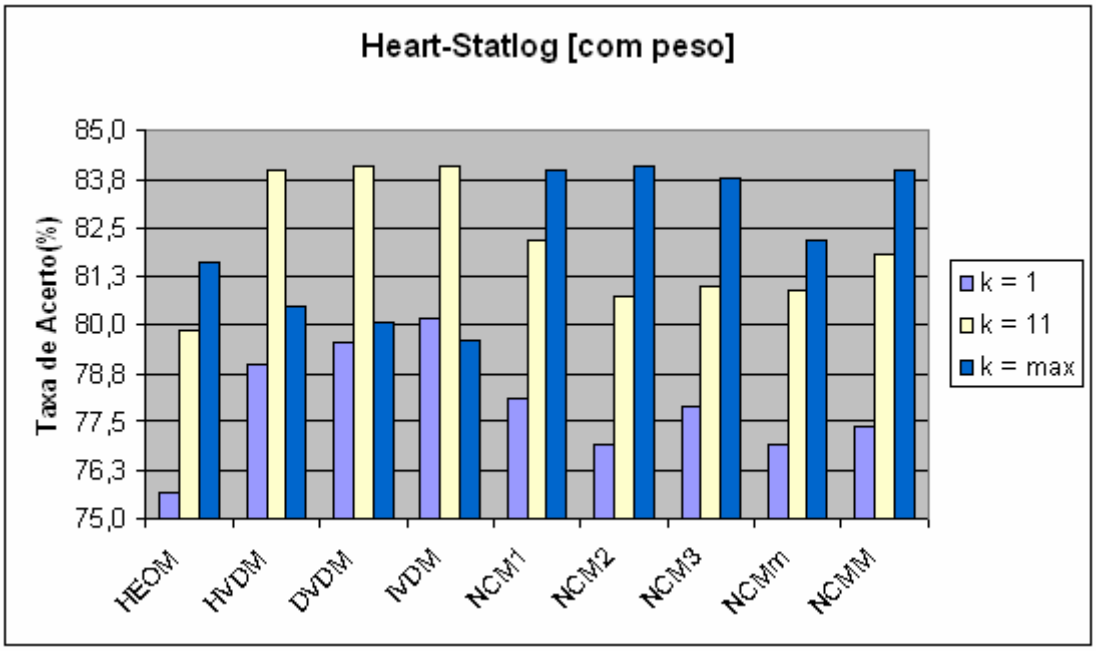

Figura 8. Taxas de acerto para a base Heart-Statlog usando a regra de classificação com peso.

4.2 Testes com RBF

O objetivo nesta seção é mostrar que a rede RBF realiza bem menos comparações do que o k-NN no momento da classificação e, apesar de mostrar uma taxa de acerto igual ou inferior, possui um desvio padrão menor. Ainda veremos que as funções de distância vão variar não só na taxa de acerto e desvio padrão, mas também no número de unidades geradas na camada intermediária.

Alguns dados novos foram medidos por se tratar de uma rede neural convém saber o número de unidades na camada intermediária, e o número de épocas para o treinamento. As legendas para os campos da tabela são:

- RBF – a taxa de acerto em (%) para a rede RBF treinada com DDA;

- (dp) – desvio padrão, na mesma unidade da taxa de acerto da coluna à sua esquerda;

– (um) – unidades médias (%), razão entre o número médio de unidades geradas na camada intermediária e o número de elementos no conjunto de treinamento e multiplicada por 100,0;

– (mU) – mínino de unidades (%), o mesmo que (um) sendo que para o caso que gerou o menor número de unidades na camada intermediária;

– (UM) – máximo de unidades (%), o mesmo que (um) sendo que para o caso que gerou o maior número de unidades na camada intermediária;

– (em) – épocas médias, número médio de épocas necessárias para a rede neural não criar mais unidades e encerrar o treinamento;

– NN – a taxa de acerto para o algoritmo NN.

Como na seção anterior podemos primeiro começar observando o resultado médio nas 14 bases utilizadas **Tabela 10**. RBFs não utiliza a função Sigmóide enquanto RBF utiliza na sua função de ativação. Como mostra a média, os resultados em ambos os casos são muito parecidos, diferenciando-se apenas em alguns poucos casos específicos. Por isso vamos utilizar somente os resultados obtidos utilizando a função Sigmóide.

Perceber-se que a taxa de acerto médio para a RBF (54,1%) é bem inferior à do NN (76,9%), contudo o desvio padrão também (2,7 e 7,6 respectivamente). E mais, o número de unidades médias é 12,5%, ou seja, para classificar uma dada instância o RBF faz somente 12,5% das comparações de distâncias feitas pelo k-NN.

Média	Acerto	(dp)	(um)	(em)
RBFs	54,0	2,8	12,2	2,3
RBF	54,1	2,7	12,5	2,3
NN	76,9	7,6	-	-

Tabela 10. RBFs (RBF sem Sigmóide); RBF (RBF com Sigmóide); NN (1-NN).

Apesar de o k-NN mostrar na média taxa de acerto bem maior que o RBF, em alguns casos esse algoritmo apresenta resultados semelhantes, o que é o caso da base **Hepatitis (Hepatt)** (**Tabela 11)** Na média da taxa de acerto do RBF é 52,5% e do NN

57,3% e os desvios padrões 3,4 e 12,1, respectivamente. Considerando apenas os casos NCM1 e NCMm o RBF apresenta-se como melhor alternativa, pois as taxas de acerto são semelhantes em ambos algoritmos contudo com um desvio padrão bem menor.

Perceber-se também que as funções de distância não só demonstraram diferentes taxas de acerto quando comparadas entre si, o que é observado também no NN, mas também apresenta variações no número de unidades geradas na camada intermediária. A média de unidades nessa camada é de 9,8%, porém fica em torno de 6% para as funções HEOM e NCMm e chega a 14% para NCMM. Nem sempre quando se tem mais neurônios na camada intermediária da RBF a taxa de acerto é maior uma vez que HVDM tem 51,1% de acerto com 10,8% de (um) e NCMm tem 53,8% com apenas 6,4% de (um).

Hepatt	**RBF**	(dp)	(um)	(mU)	(MU)	(em)	**NN**	(dp)
HEOM	**50,2**	3,8	6,3	2,9	13,6	2,1	**59,9**	12,5
HVDM	**51,1**	3,4	10,8	3,6	22,1	2,3	**58,8**	11,4
DVDM	**51,6**	3,6	8,7	2,1	19,3	2,2	**58,3**	11,2
IVDM	**51,4**	3,1	9,5	3,6	23,6	2,3	**59,0**	11,8
NCM1	**51,9**	3,5	9,2	2,9	27,1	2,1	**53,2**	11,8
NCM2	**53,9**	3,2	11,7	2,9	40,0	2,1	**57,8**	12,6
NCM3	**54,1**	3,3	11,5	2,9	40,0	2,1	**57,6**	12,8
NCMm	**53,8**	3,4	6,4	2,1	17,9	2,1	**54,7**	12,6
NCMM	**54,1**	3,3	14,0	3,6	42,1	2,0	**56,3**	12,1
Média	**52,5**	**3,4**	**9,8**	**2,9**	**27,3**	**2,1**	**57,3**	**12,1**

Tabela 11. Dados do RBF aplicado à base Hepatitis.

Outros casos além de Hepatitis apresentaram resultados na RBF semelhantes ao NN como é o caso de Breast-Cancer, Heart-Statlog e Zoo. Um dos casos onde o RBF é bem pior que NN é **Primary Tumor (PrimT) (Tabela 12).** Contudo o desvio padrão continua sendo bem menor, também. E o número de (um) apesar de ser o mais alto entre todas as bases, 51,5 %, ainda assim é uma boa característica, pois reduz o número de comparações. Vale destacar que bases como Wine e Iris tiveram em média 4% de (um).

Utilizar a função de ativação com ou sem Sigmóide pode também interferir nos resultados, como mostrado na base **Íris (Tabela 13)** que têm os melhores resultados com a função de ativação mais simples (sem Sigmóide). Não só a maior taxa de acerto como também o menor desvio padrão e menos neurônios na camada intermediária. Já

base **Breast-Cancer** possui os melhores resultado para algumas funções de distância usando Sigmóide e para outras distâncias sem usá-la. Existem bases cujos melhores resultados estão sempre usando Sigmóide.

PrimT	**RBF**	(dp)	(um)	(mU)	(MU)	(em)	**NN**	(dp)
HEOM	**21,7**	1,4	47,1	36,0	59,3	2,9	**35,5**	8,8
HVDM	**23,7**	1,3	59,5	53,6	64,7	3,1	**36,1**	8,9
DVDM	**23,7**	1,3	59,5	53,6	64,7	3,1	**36,1**	8,9
IVDM	**23,7**	1,3	59,5	53,6	64,7	3,1	**36,1**	8,9
NCM1	**21,8**	1,8	47,6	38,9	59,5	2,9	**33,7**	8,2
NCM2	**21,8**	1,8	47,6	38,9	59,5	2,9	**33,7**	8,2
NCM3	**21,8**	1,8	47,6	38,9	59,5	2,9	**33,7**	8,2
NCMm	**21,8**	1,8	47,6	38,9	59,5	2,9	**33,7**	8,2
NCMM	**21,8**	1,8	47,6	38,9	59,5	2,9	**33,7**	8,2
Média	**22,4**	**1,6**	**51,5**	**43,5**	**61,2**	**2,9**	**34,7**	**8,5**

Tabela 12. Dados do RBF aplicado à base Primary Tumor.

Iris	**RBFs**	(dp)	(um)	**RBF**	(dp)	(um)
HEOM	**85,2**	2,9	3,9	**79,5**	3,3	4,4
HVDM	**85,2**	2,9	3,9	**79,5**	3,3	4,4
DVDM	**77,8**	3,8	3,2	**74,1**	4,5	3,5
IVDM	**75,6**	4,0	3,1	**74,2**	3,8	3,1
NCM1	**68,6**	4,1	3,1	**63,2**	3,7	3,6
NCM2	**72,3**	3,7	3,2	**65,1**	3,8	3,7
NCM3	**72,7**	3,8	3,2	**65,6**	3,8	3,7
NCMm	**80,3**	2,9	3,4	**74,4**	3,9	3,9
NCMM	**68,0**	4,1	3,0	**61,6**	3,7	3,5
Média	**76,2**	**3,5**	**3,3**	**70,8**	**3,8**	**3,7**

Tabela 13. Dados do RBF aplicado à base Iris.
Comparando RBFs (sem Sigmóide) e RBF (com Sigmóide).

As tabelas completas, para todas as bases, usando função de ativação com e sem Sigmóide estão em anexo. Com todos os dados medidos, como acima, e com o acerto e desvio padrão do NN.

4.3 Comparando funções de distâncias no RBF e no NN

Na seção 4.1 é constatado que as funções de distância podem variar bastante seu comportamento no k-NN para valores diferentes de k. Ainda foi visto para o k-NN que algumas funções de distância apresentam melhores resultados usando a regra de classificação com peso e outras usando a regra de energia ou sem peso.

O objetivo dessa seção é mostrar que em algoritmos distintos, a resposta de cada função pode variar, ou seja, nem sempre uma função que apresentou os melhores resultados no k-NN, para a mesma base, vai ser a melhor quando empregada no RBF. Para tanto vamos comparar a RBF (usando Sigmóide) com o NN.

Os gráficos foram construídos da seguinte maneira: para cada algoritmo (RBF e NN) constam nove barras, uma para cada função de distância. O valor de cada barra é a taxa de acerto daquela função dividida pela taxa de acerto máxima no mesmo algoritmo.

Para a base **Primary Tumor** as funções de distância apresentam comportamento semelhante em ambos os algoritmos. Como já comentado: essa base só possui atributos categóricos e pode-se considerar que só foram usadas três funções de distância. Vê-se claramente que em todos os casos apresentou seu máximo com VDM e foi sempre pior com NCM. A base **Wine**, que possui somente atributos numéricos, difere nas funções NCM, que apresentaram as maiores taxas de acerto no k-NN e muito inferiores na RBF. Para as demais funções de distância, Wine, apresenta comportamento semelhante. Já **Hepatitis** apresenta os melhores resultados em RBF usando NCM e no k-NN os piores resultados são NCM. (**Figuras 9, 10 e 11**).

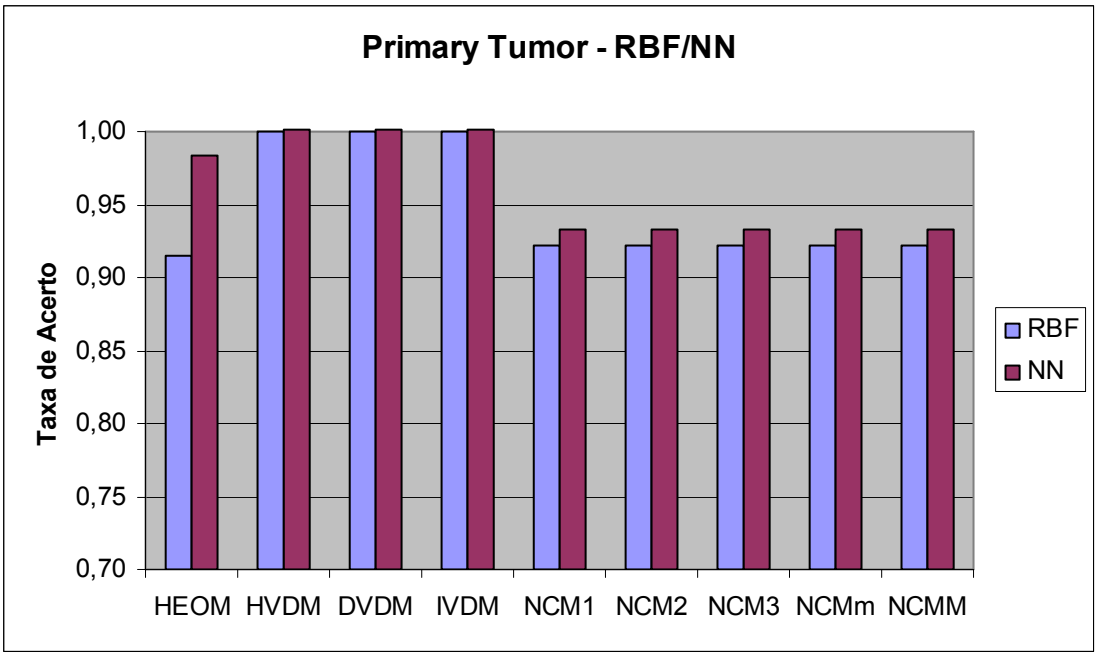

Figura 9. Comparação do comportamento das funções de distância no NN e na rede RBF para a base Primary Tumor.

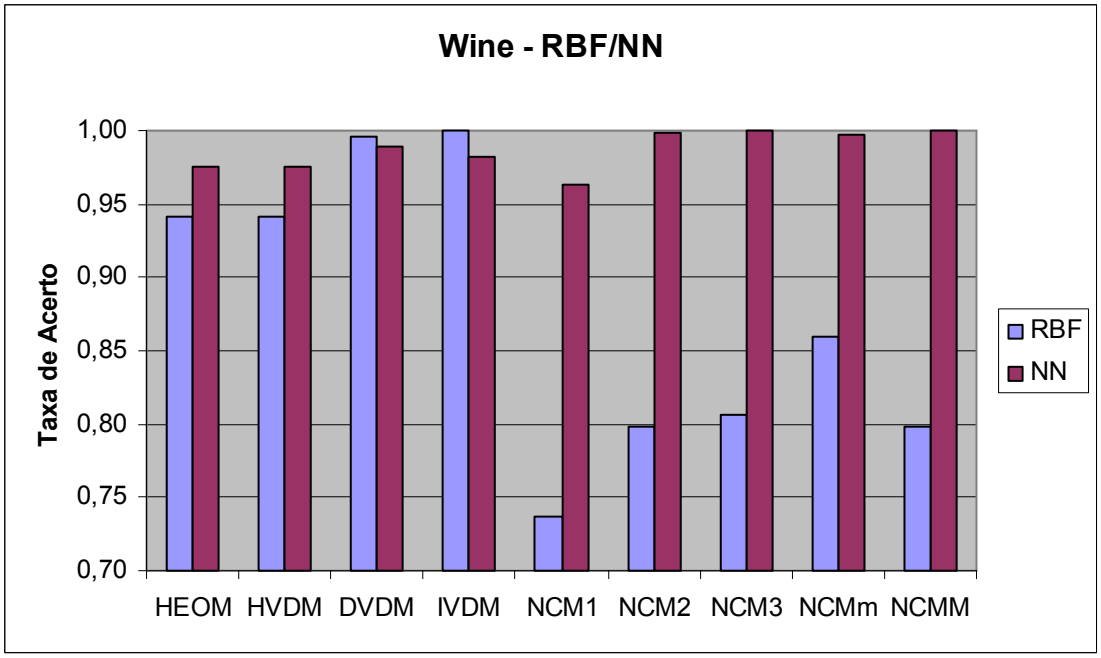

Figura 10. Comparação do comportamento das funções de distância no NN e na rede RBF para a base Wine.

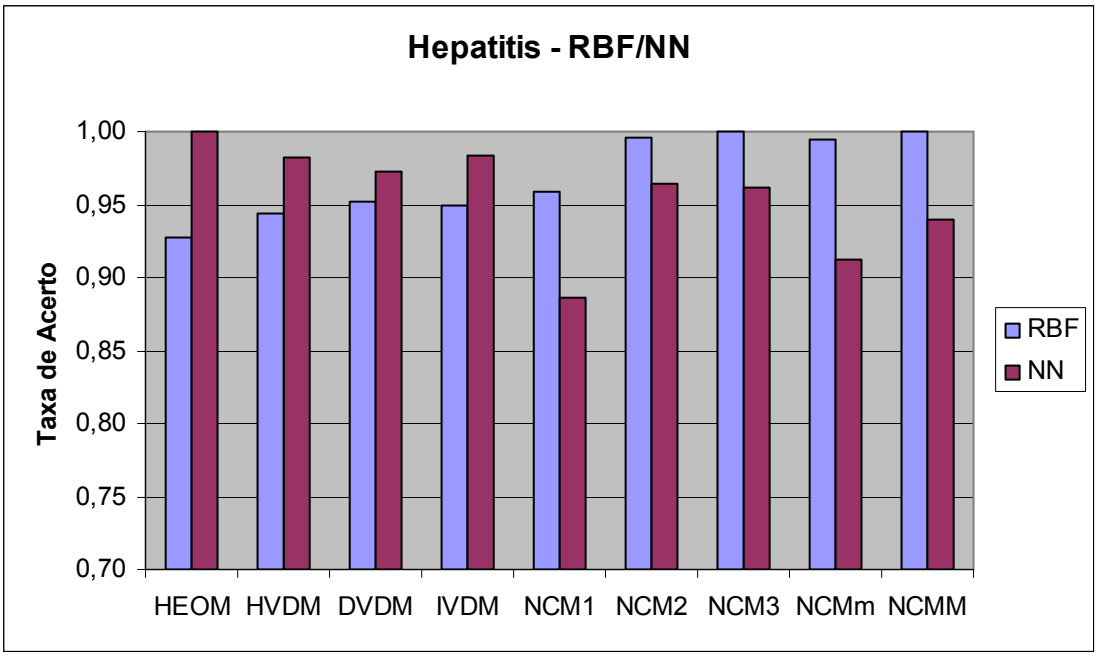

Figura 11. Comparação do comportamento das funções de distância no NN e na rede RBF para a base Hepatitis.

Esses três gráficos representam muito bem todos os estudos. E servem para demonstrar que a função de distância varia os resultados não só dependendo da base de dados sobre a qual efetua suas computações. Também o algoritmo influencia na hora de escolher uma distância mais adequada.

5. Conclusão

Uma série de funções de distância que podem ser usadas em algoritmos de classificação. Algumas dessas distâncias como NCM (e variações) e HEOM não se restringem a essa tarefa e podem ser aplicadas a outros problemas que empreguem o cálculo da distância. NCM é uma medida de distância nova, proposta em 2006 por Wang. Foram propostas aqui algumas pequenas alterações nessa função, com o objetivo inicial de transformá-la numa função em simétrica, seguindo os conceitos de métrica de [Theodoridis]. Essas pequenas alterações geraram novas formas de calcular distância que apresentam comportamento tão distintos que podem se consideradas como funções completamente diferentes.

Da literatura foram listadas as funções HEOM, DVDM, IVDM e NCM1. Em seguida propostas NCM2, NCM3, NCMm e NCMM. Foram realizados testes, usando essas nove distâncias, no classificador k-NN e na rede neural RBF treinada com DDA. Os resultados desses testes mostraram que, dependendo da situação, alguma função de distância se adeqüa melhor. Essas situações podem ser: algoritmos distintos; ou configurações distintas de um mesmo algoritmo, como por exemplo, o valor de k ou a regra de classificação no k-NN e a função de ativação na RBF. Também foi constado que a base de dados tem uma forte influência na escolha da forma de calcular a distância, devido à sua distribuição de classes, aos tipos de atributos e outras características não obvias. Em suma: dada uma base de dados, um algoritmo de classificação e uma configuração fixa para esse algoritmo é possível escolher uma função de distância que obtenha melhores resultados. Se uma dessas três partes mudar possivelmente outra função de distância será mais apropriada, sob os mesmo critérios, do que a escolhida anteriormente.

Como proposta para trabalhos futuros fica testar essas funções no k-NN para valores de $k < 6$, uma vez foi levantada a questão de algumas bases apresentaria seus melhores resultados nessa faixa. Testar as funções de distância enumeradas aqui no algoritmo k-NN com peso proposto em [Paredes*] que estava na proposta inicial, mas que não obteve sucesso na implementação devido a algumas informações erradas descritas no artigo e descobertas tardiamente. Também fica como proposta testar essas

funções em outros algoritmos que desempenham outras atividades diferente da classificação, como, por exemplo, algoritmos de agrupamento.

Ainda fica proposta a idéia de se montar uma função de distância que combine o melhor de cada uma vista aqui. Isto é, sob as mesmas três condições pré-dispostas – uma mesma base de dados, um algoritmo de classificação e uma configuração fixa para esse algoritmo – compor uma função de distância onde a forma de calcular a distância pode ser completamente diferente para cada posição i do vetor de atributos. Possivelmente pode-se usar algoritmos genéticos para se construir essa função.

Referências

[Berthold; Diamond 95] Michael R. Berthold, Jay Diamond, “Boosting the Performance of RBF Networks with Dynamic Decay Adjustment”, in G, Tesaurus, D. Touretzky and T.Leen (eds): Advances in Neural Information Processing, 7, 1995

[Healey 1990] Joseph Healey, “Statistics: A Tool for Social Research”, Wadsworth Publishing Company, 1990.

[Kohavi 95] Ron Kohavi, "A Study of Cross-Validation and Bootstrapfor Accuracy Estimation and Model Selection", International Joint Conference on Artificial Intelligence (IJCAI), 1995

[Mitchell 97] Tom Mitchell, "Machine Learning", McGraw-Hill, 1997.

[Oliveira *et al* 05] Adriano L. I. Oliveira, B. J .M. Melo, S. R. L. Meira, “Integrated method for construtive training of radial basis function network”, ELECTRONICS LETTERS,vol. 41, No. 7, March 2005.

[Paredes; Vidal 06] Roberto Paredes, Enrique Vidal, “Learning Weighted Metrics to Miniminize Nearest-Neighbor Classification Error”, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.28, no.7, pp. 1100-1110, july 2006

[Stanfill; Waltz 1986] C. Stanfill, D. Waltz, “Toward Memory-Based Reasoning”, Comm. ACM, vol 29, pp. 1213-1229, 1986

[Theodoridis; Koutroumbas 03] Sergios Theodoridis, Konstantinos Koutroumbas, "Pattern Recognition (second edition)”, ACADEMIC PRESS (An imprint of Elsevier), 2003

[UCI 07] UCI Machine Learning Repository, http://www.ics.uci.edu/~mlearn/MLRepository.html, acesso em 2007

[Wang 06] Hui Wang, “Nearest Neighbor by Neighborhood Counting”, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.28, no.6, pp. 942-953, june 2006

[Wilson; Martinez 97] D. R. Wilson and T. R. Martinez, “Improved Heterogeneous Distance Functions”, J. Artificial Intelligence Research, vol.6, pp.1-34,1997.

[Yamada *et al* 06] T. Yamada, K. Yamashita, N. Ishii, K. Iwata, "Text Classification by Combining Different Distance Functions with Weights", Seventh ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, 2006. SNPD 2006. Volume, Issue, 19-20 June 2006, pp. 85 – 90