UNIVERSIDADE DE SÃO PAULO
INSTITUTO DE CIÊNCIAS MATEMÁTICAS E DE COMPUTAÇÃO
http://www.icmc.usp.br
Departamento de Ciências de Computação

SCC-630 - Capítulo 11
Classificação de Atributos

João Luís Garcia Rosa[1]

[1]Departamento de Ciências de Computação
Instituto de Ciências Matemáticas e de Computação
Universidade de São Paulo - São Carlos
joaoluis@icmc.usp.br

2011

Agradecimento

Agradeço à Profa. Maria Carolina Monard, que gentilmente permitiu que eu usasse seus slides [2] para preparação deste capítulo.

Sumário

Material do Eamonn Keogh

Os próximos 51 slides contêm material do Prof. Eamonn Keogh [1], com adaptação da Profa. Maria Carolina Monard.

Fair Use Agreement

This agreement covers the use of all slides on this CD-Rom, please read carefully.

- You may freely use these slides for teaching, if
  - You send me an email telling me the class number/ university in advance.
  - My name and email address appears on the first slide (if you are using all or most of the slides), or on each slide (if you are just taking a few slides).
- You may freely use these slides for a conference presentation, if
  - You send me an email telling me the conference name in advance.
  - My name appears on each slide you use.

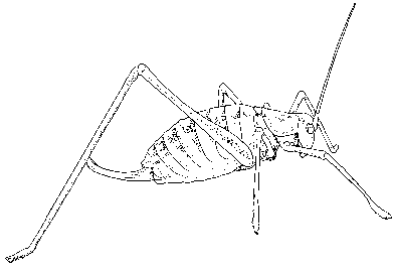
- You may not use these slides for tutorials, or in a published work (tech report/ conference paper/ thesis/ journal etc). If you wish to do this, email me first, it is highly likely I will grant you permission.

(c) Eamonn Keogh, eamonn@cs.ucr.edu

O problema de classificação

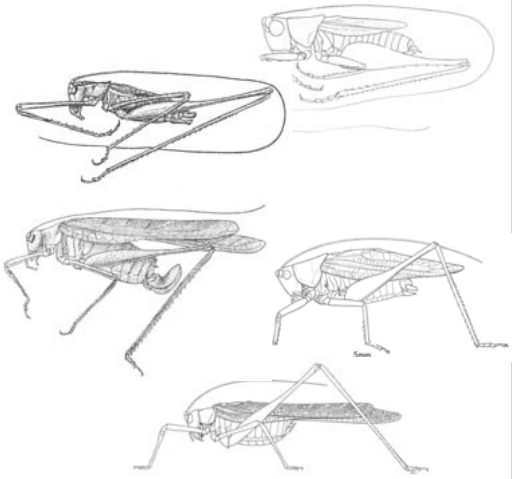
(definição informal)

Dada uma coleção de dados detalhados, neste caso 5 exemplos de **Esperança** e 5 do **Gafanhoto**, decida a qual tipo de inseto o exemplo não rotulado pertence.
Obs: **Esperança** : tipo de gafanhoto verde.

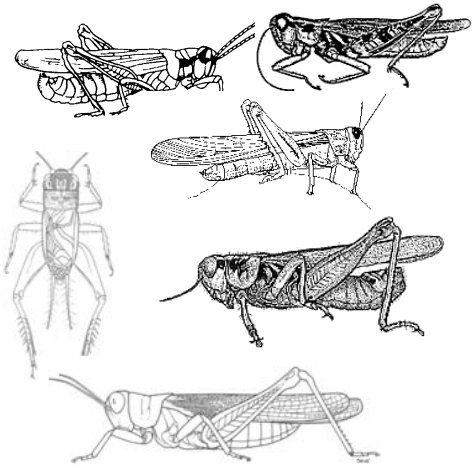


Esperança ou **Gafanhoto**?

Esperança



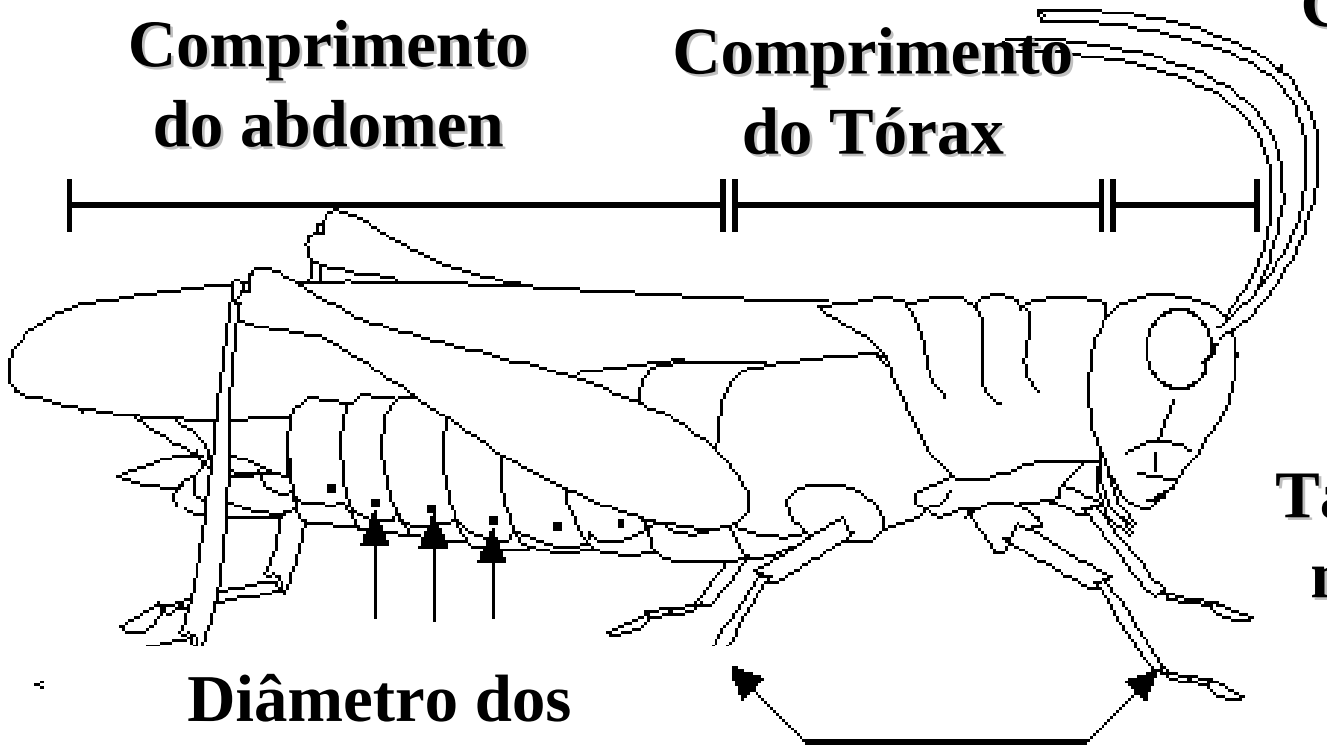
Gafanhoto



Para qualquer domínio de interesse podemos medir *características*

Cor {Verde, Marrom, Cinza, Outra}

Tem asas?



Podemos armazenar as *características* em bases de dados

O problema de classificação agora pode ser expresso da seguinte forma:

• Dada uma base de treinamento(**Minha_Coleção**), prediga o rótulo da **classe dos exemplos ainda não vistos**

Minha_Coleção

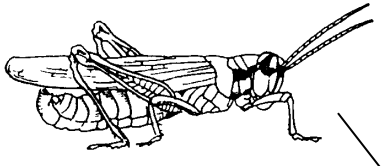
ID do inseto	Comp. do abdômen	Comp. das antenas	Classe do inseto
1	2.7	5.5	**Gafanhoto**
2	8.0	9.1	**Esperança**
3	0.9	4.7	**Gafanhoto**
4	1.1	3.1	**Gafanhoto**
5	5.4	8.5	**Esperança**
6	2.9	1.9	**Gafanhoto**
7	6.1	6.6	**Esperança**
8	0.5	1.0	**Gafanhoto**
9	8.3	6.6	**Esperança**
10	8.1	4.7	**Esperança**

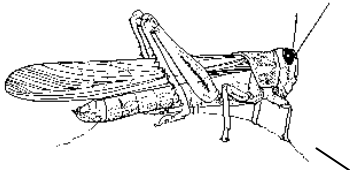
Exemplo não visto =

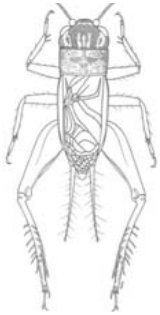
11	5.1	7.0	???????

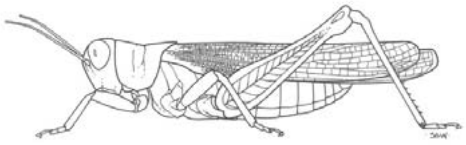
Gafanhoto

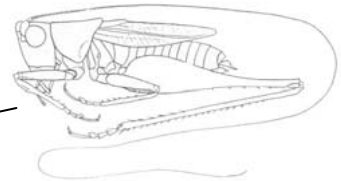
Esperança

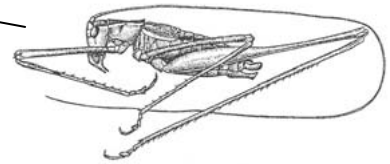


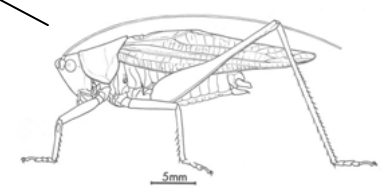


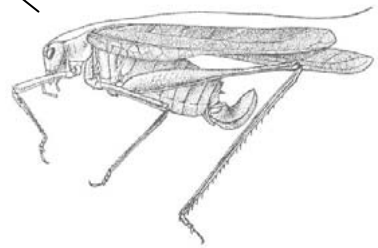


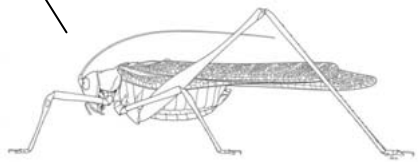


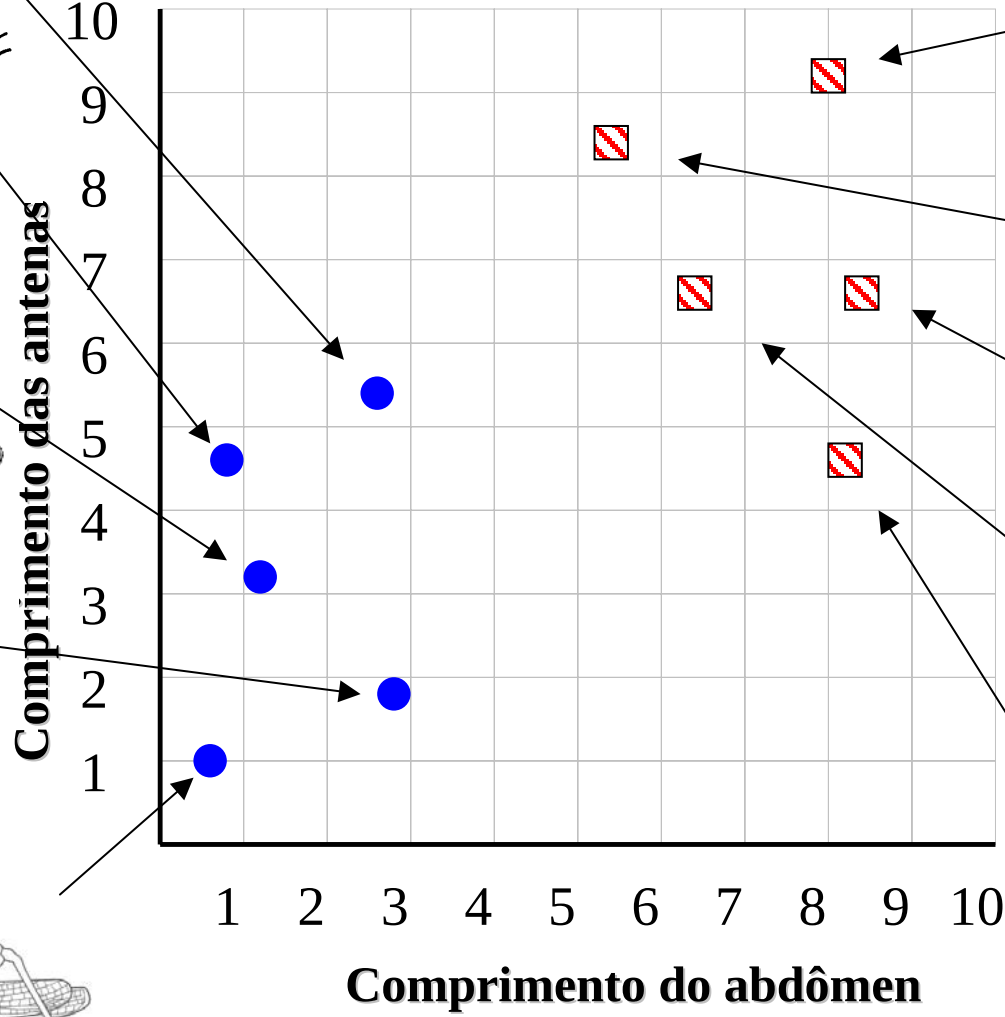




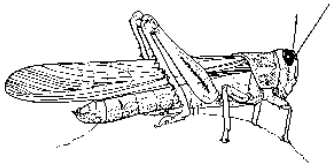




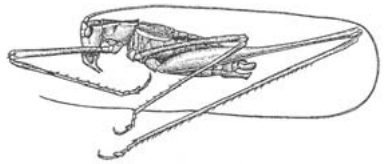




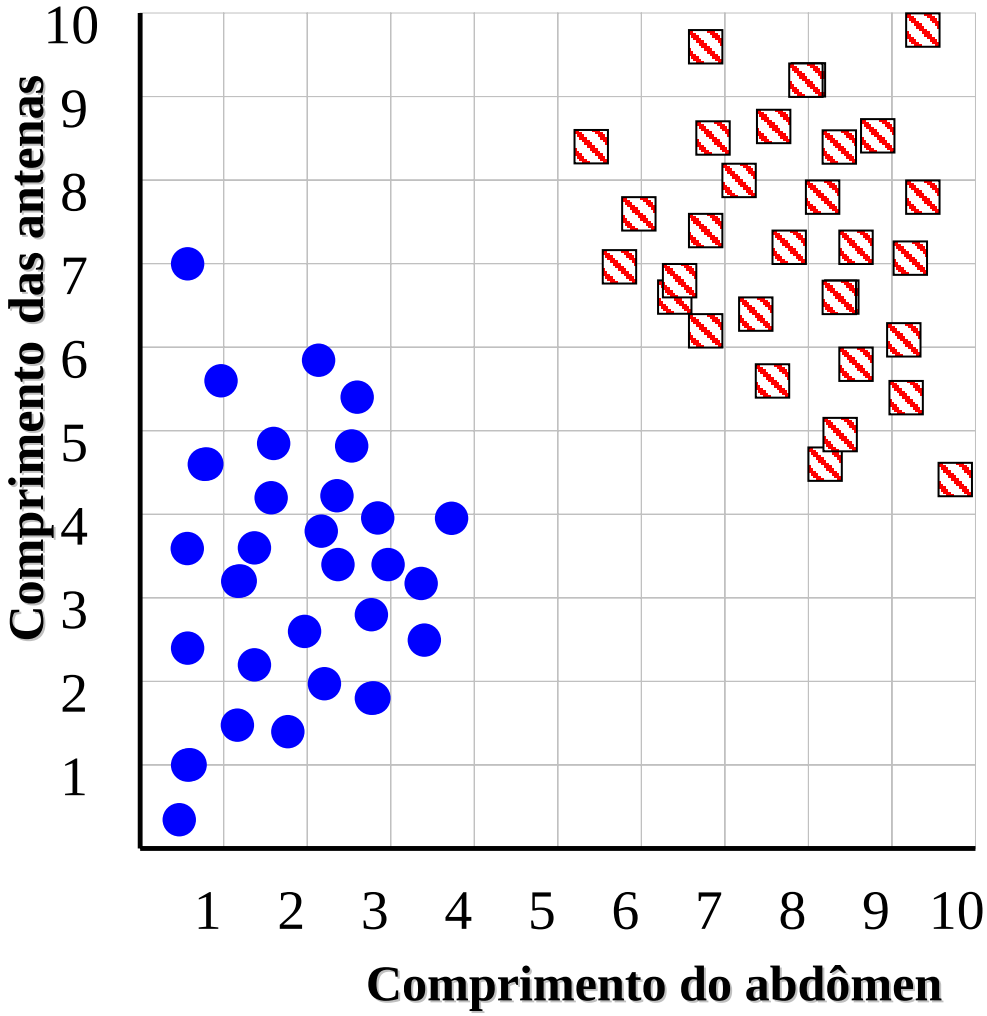
Gafanhoto



Esperança

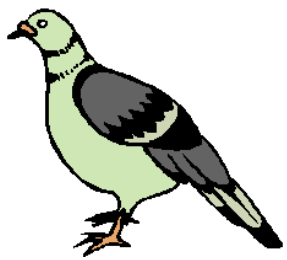


Também utilizaremos esta base de dados maior para motivação …



Cada um destes objetos de dados é chamado de…

- exemplar
- exemplo (de treinamento)
- instância
- tupla



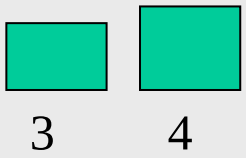
Voltaremos ao slide anterior em dois minutos. Enquanto isso vamos jogar um joguinho rápido.

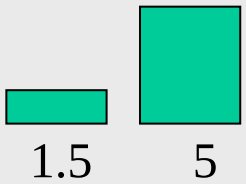
Vou mostrar a vocês alguns problemas de classificação que foram mostrados a pombos!

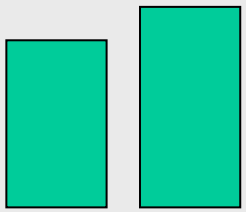
Vamos ver se você é tão esperto quanto um pombo!

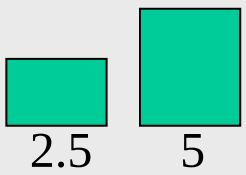
Problema do Pombo 1

Exemplos da classe A

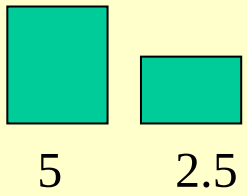


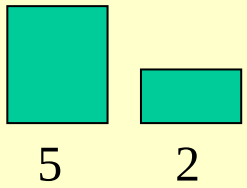


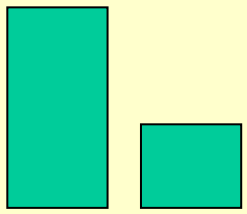


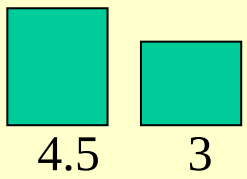


Exemplos da classe B



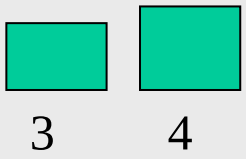


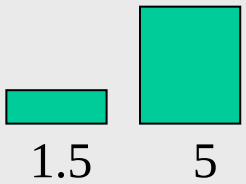


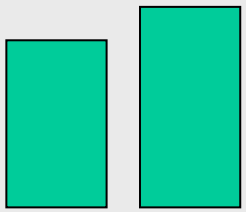


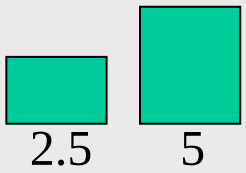
Problema do Pombo 1

Exemplos da classe A

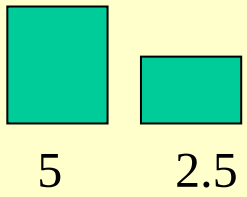


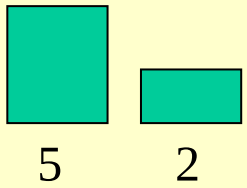


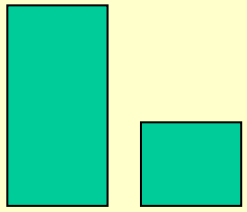


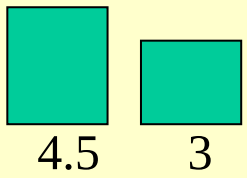


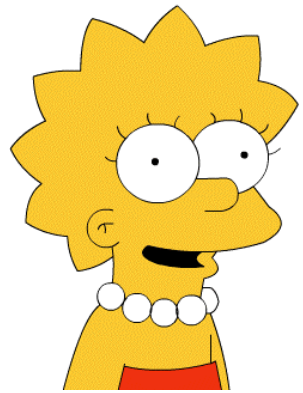
Exemplos da classe B



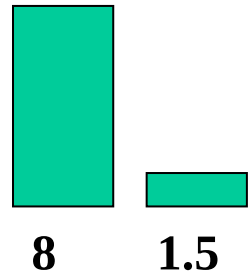




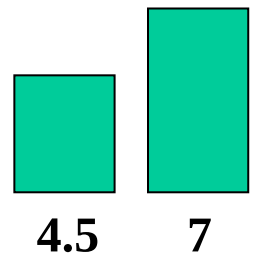




De qual classe é este objeto?

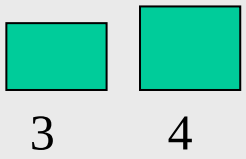


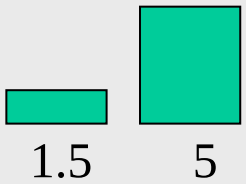
Que tal este, A ou B?

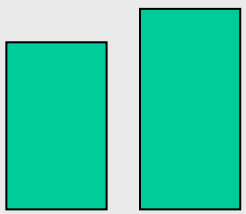


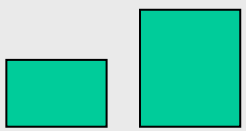
Problema do Pombo 1

Exemplos da classe A

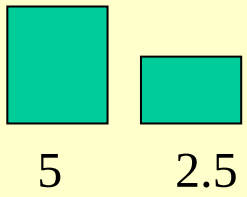


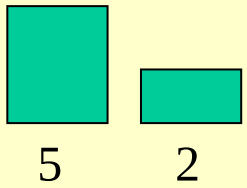


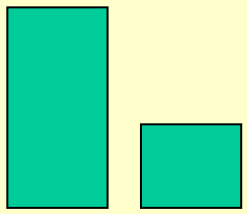


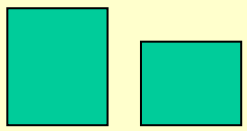


Exemplos da classe B

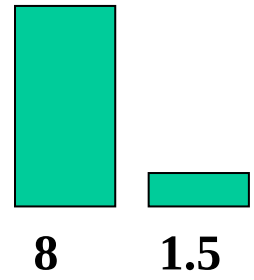








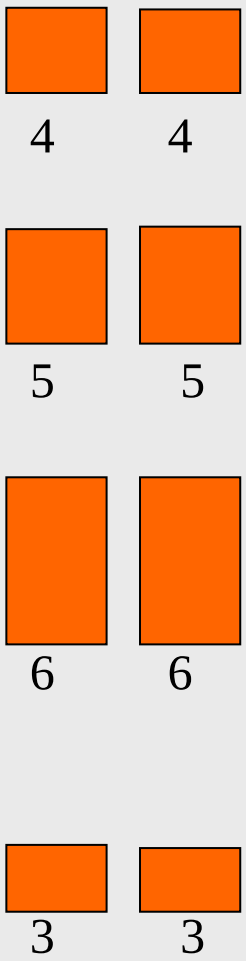
Este é um B!



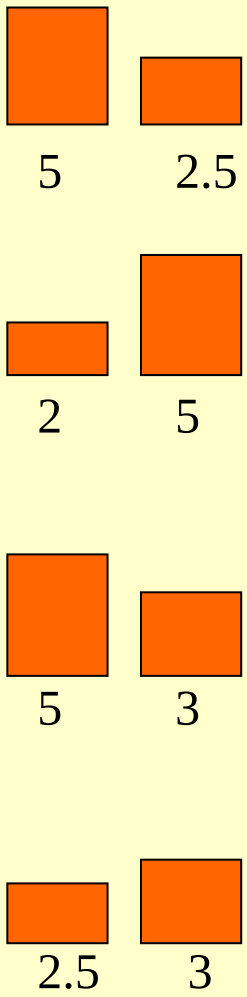
Eis a regra. Se a barra esquerda é menor que a direita, é um A, caso contrário é um B.

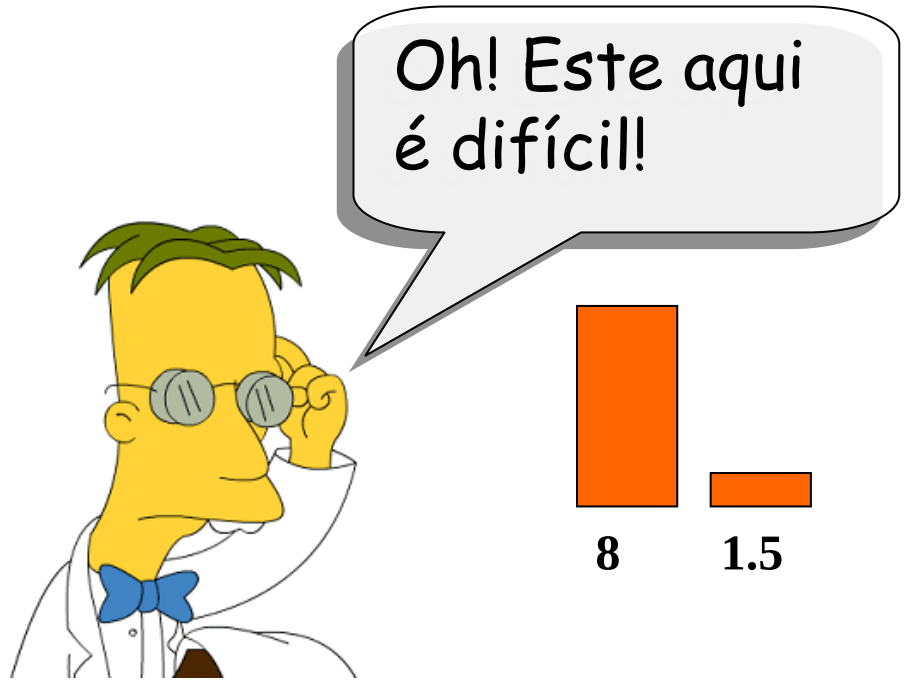
Problema do Pombo 2

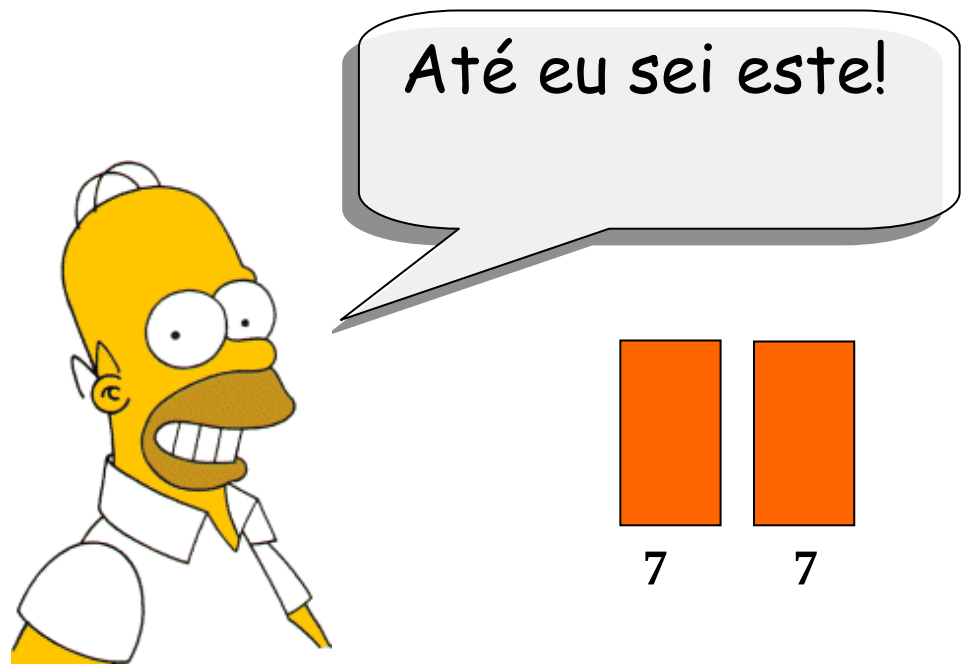
Exemplos da classe A



Exemplos da classe B

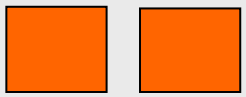






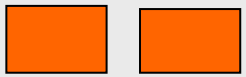
Problema do Pombo 2

Exemplos da classe A

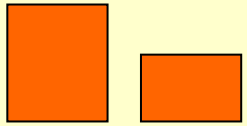

4 4

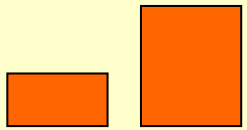

5 5

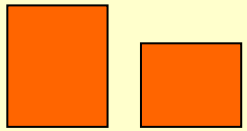

6 6

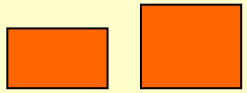

3 3

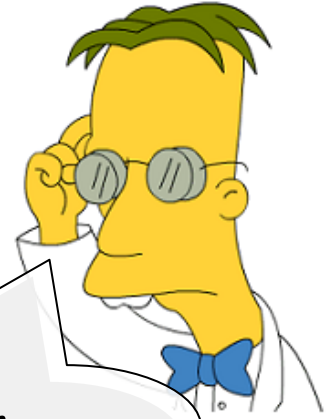
Exemplos da classe B


5 2.5


2 5

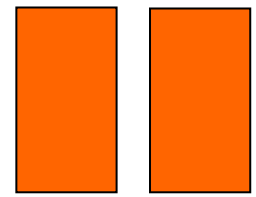

5 3


2.5 3



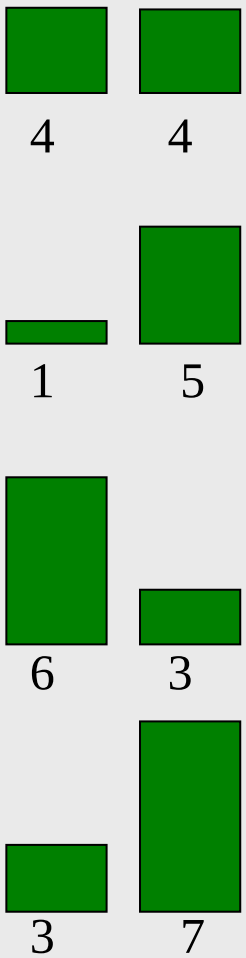
A regra é: se duas barras são iguais em tamanho é um A. Caso contrário é um B.

Então este é um A.

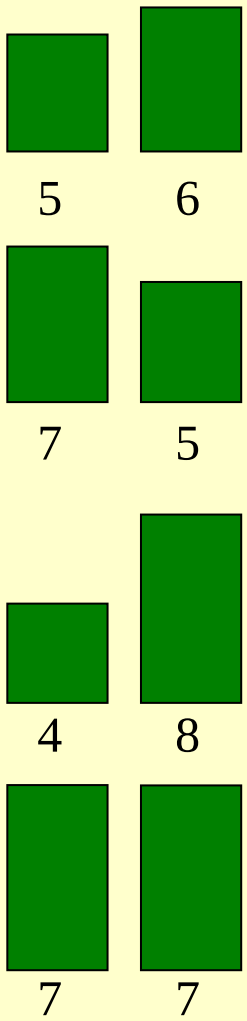

7 7

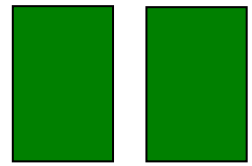
Problema do Pombo 3

Exemplos da classe A



Exemplos da classe B

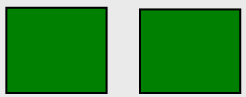


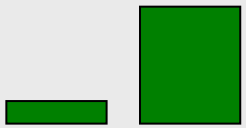


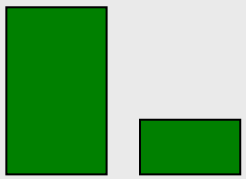
Este é muito difícil!
Qual é este, A ou B?

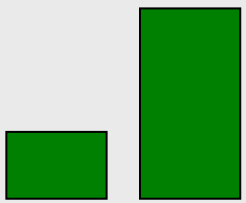
Problema do Pombo 3

Exemplos da classe A

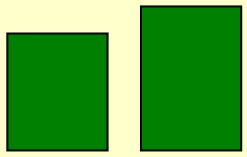

4 4

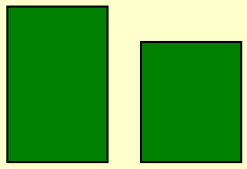

1 5

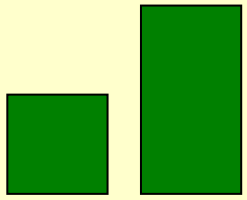

6 3

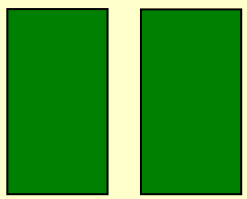

3 7

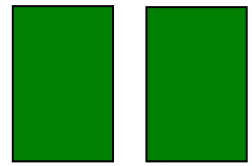
Exemplos da classe B


5 6


7 5


4 8

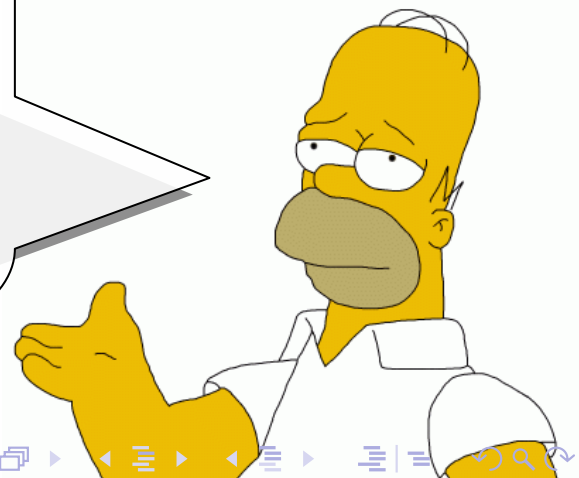

7 7


6 6

A regra é a seguinte, se o quadrado da soma das duas barras é menor ou igual a 100, é um A. Caso contrário é um B.

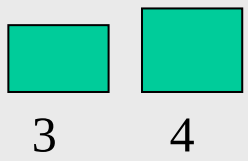
Por que gastamos tanto tempo com este joguinho?

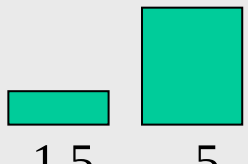
Porque queriamos mostrar que quase todos os problemas de classificação tem uma interpretação geométrica. Confira os próximos 3 slides...

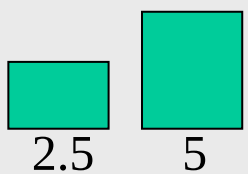


Problema do Pombo 1

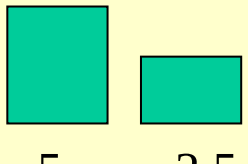
Exemplos da classe A

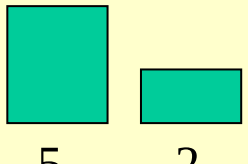


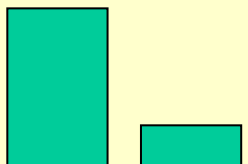


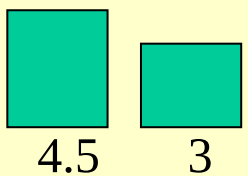


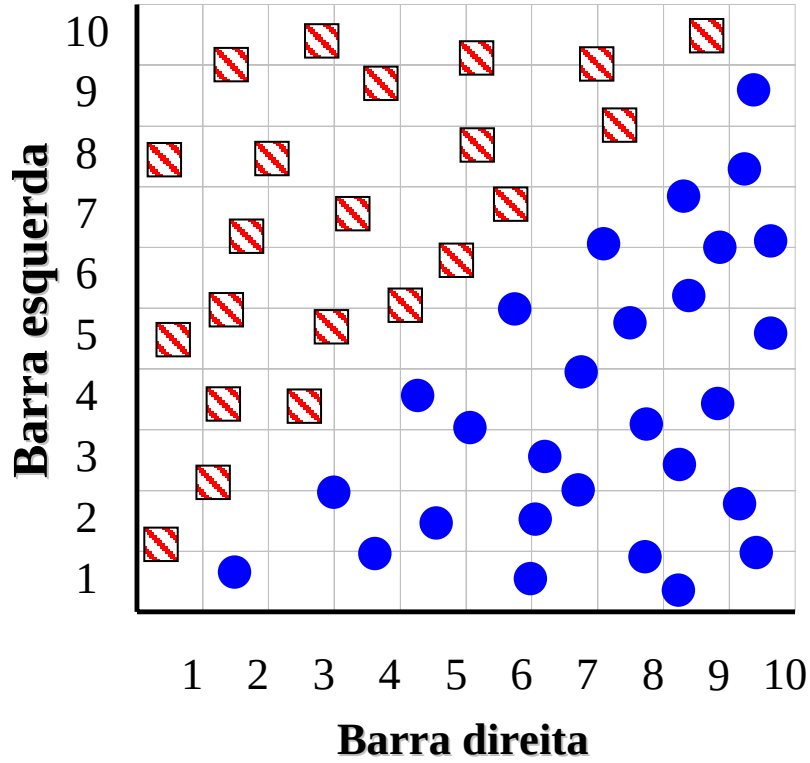
Exemplos da classe B







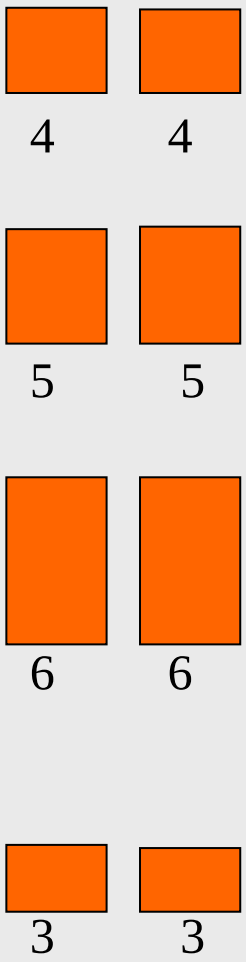




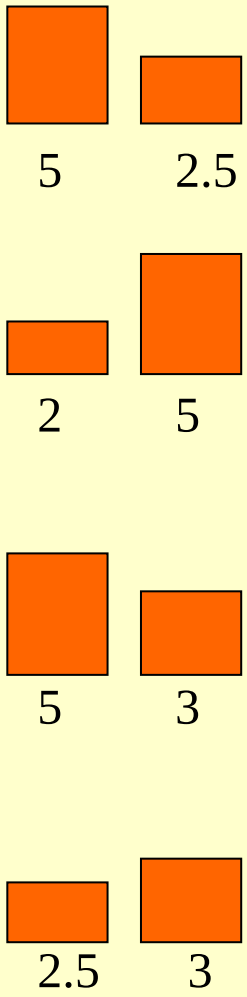
Eis a regra novamente. Se a barra esquerda é menor que a direita, é um A, caso contrário é um B.

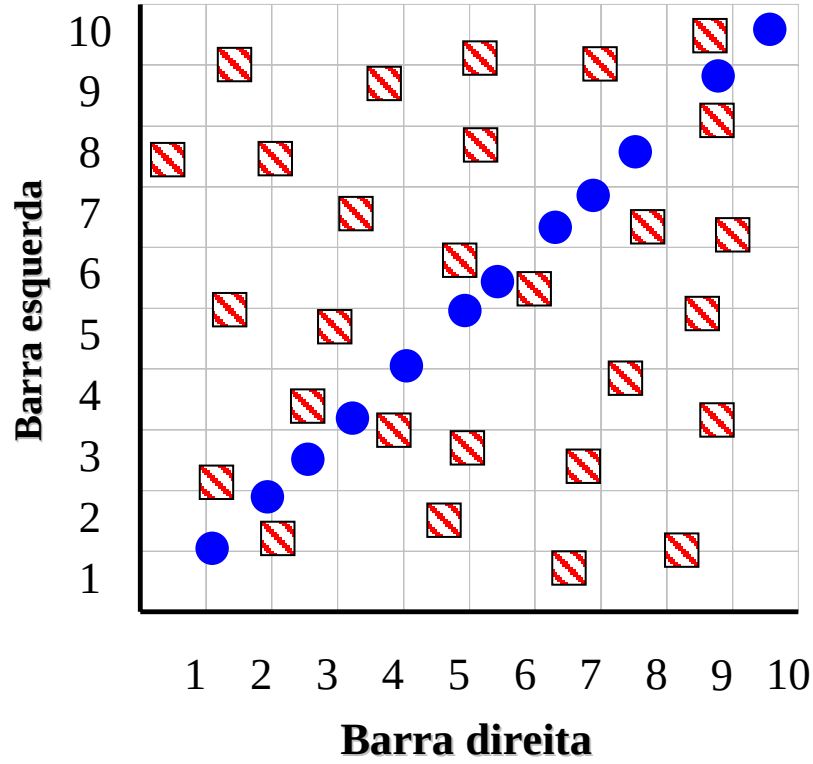
Problema do Pombo 2

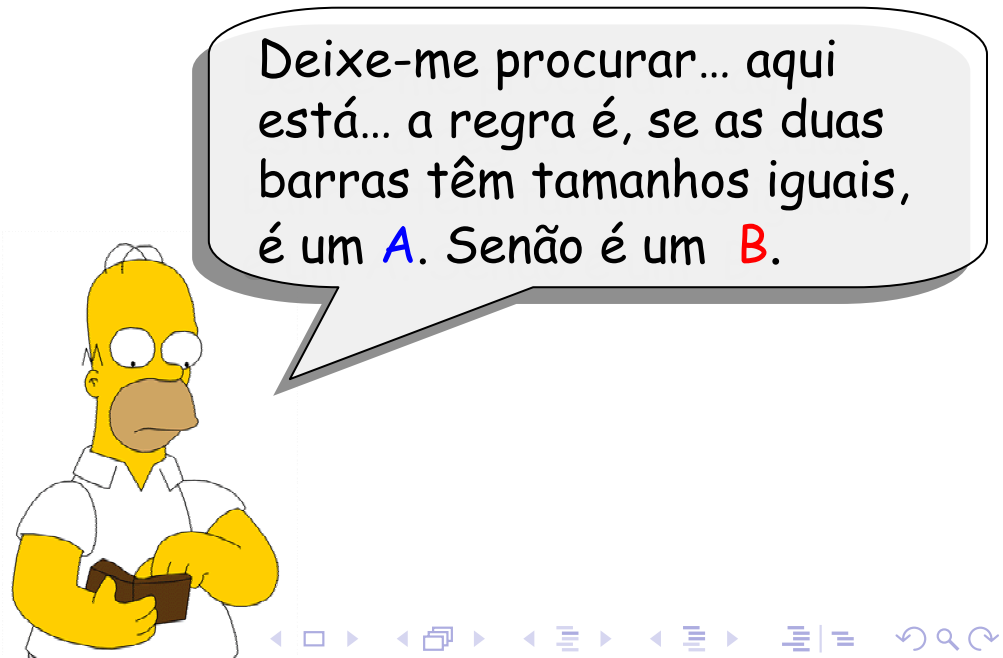
Exemplos da classe A



Exemplos da classe B

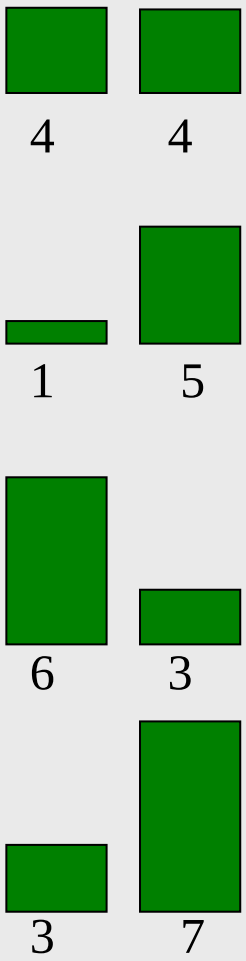




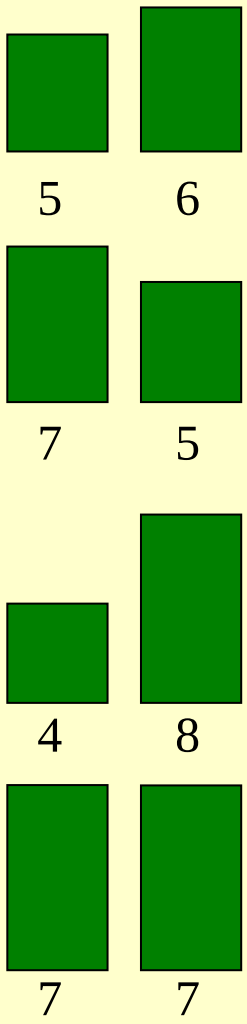


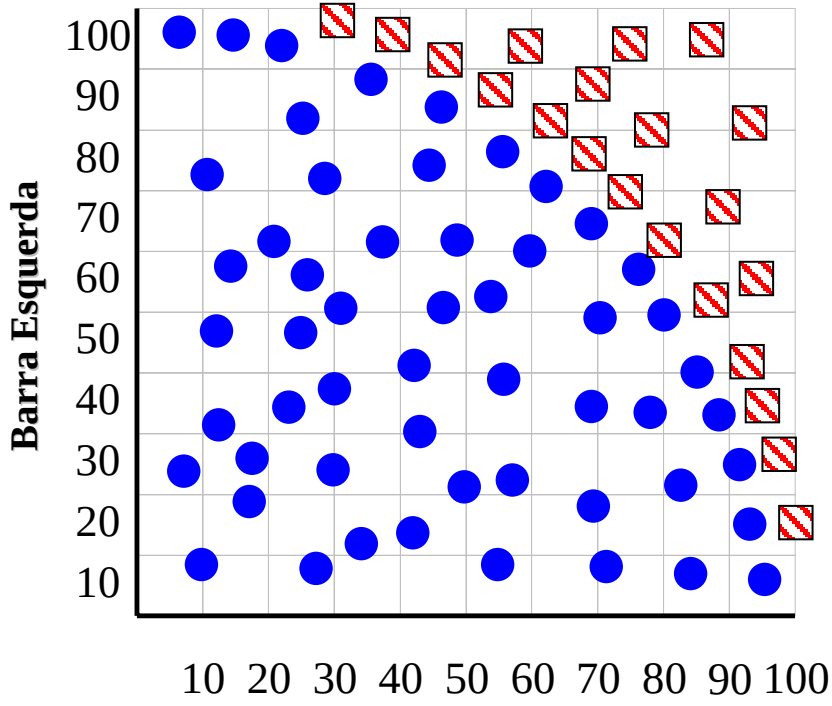
Problema do Pombo 3

Exemplos da classe A



Exemplos da classe B

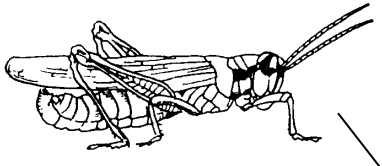


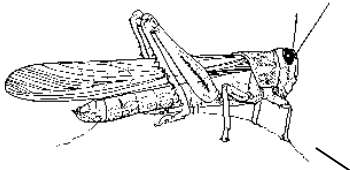


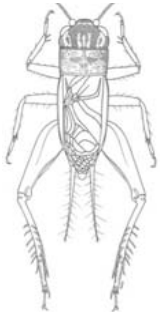
A regra novamente:
Se o quadrado da soma das duas barras é menor ou igual a 100, é um A. Senão é um B.

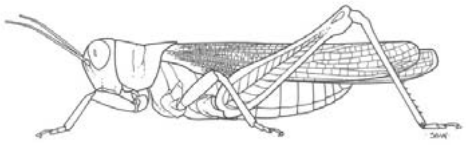
Gafanhoto

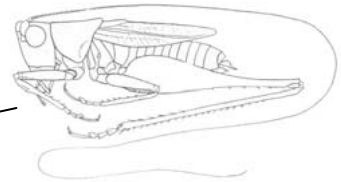
Esperança

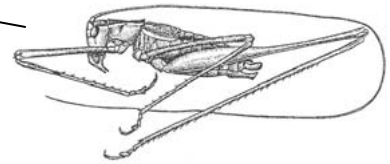


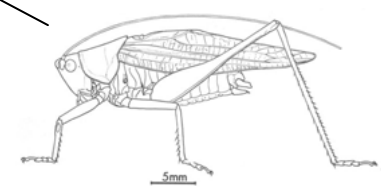


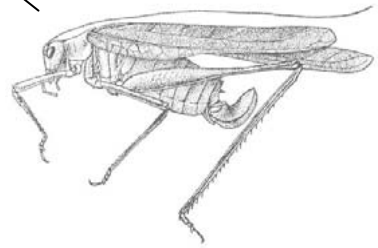


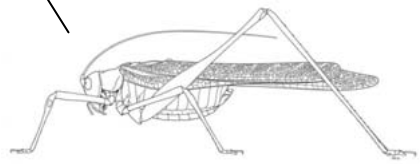


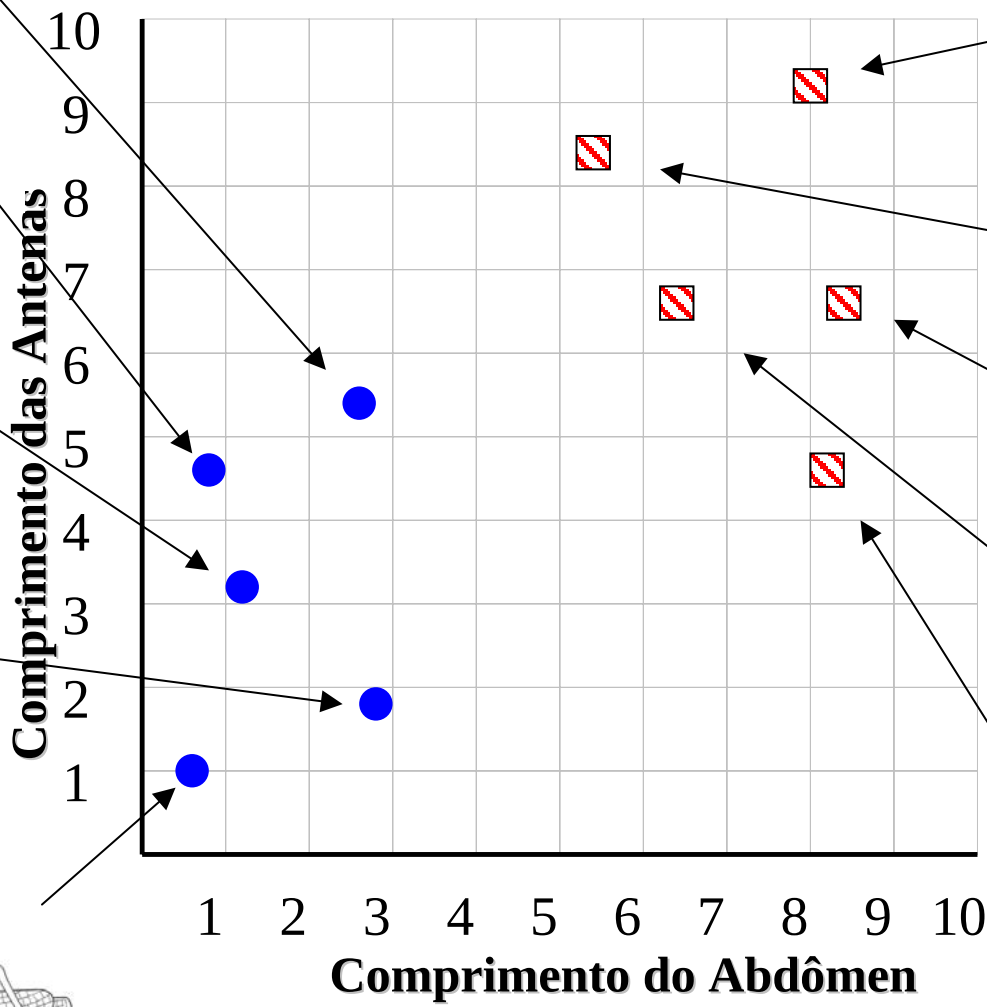






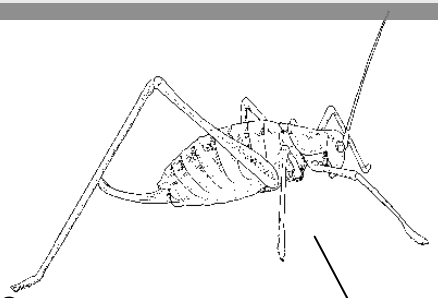


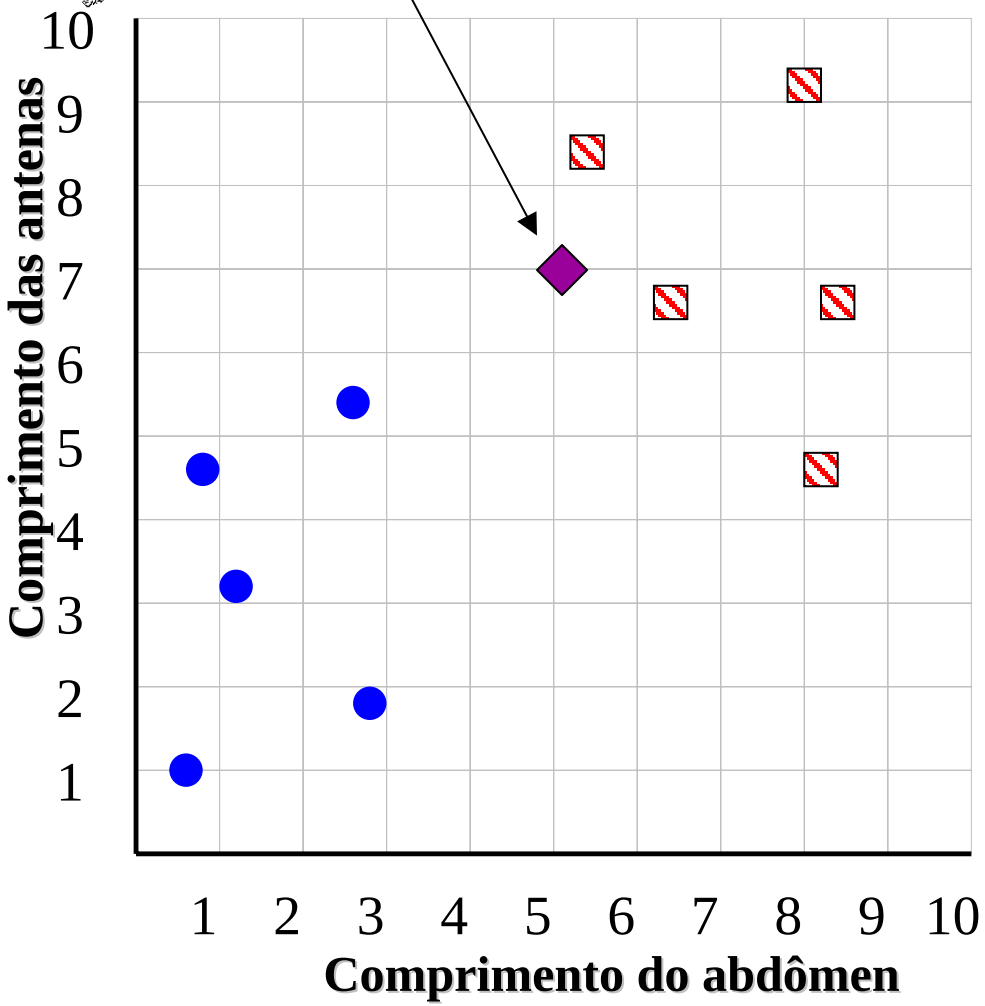




Exemplo não visto antes =

11	**5.1**	**7.0**	???????





- Podemos "projetar" o **exemplo não visto antes** dentro do mesmo espaço que a base de dados.

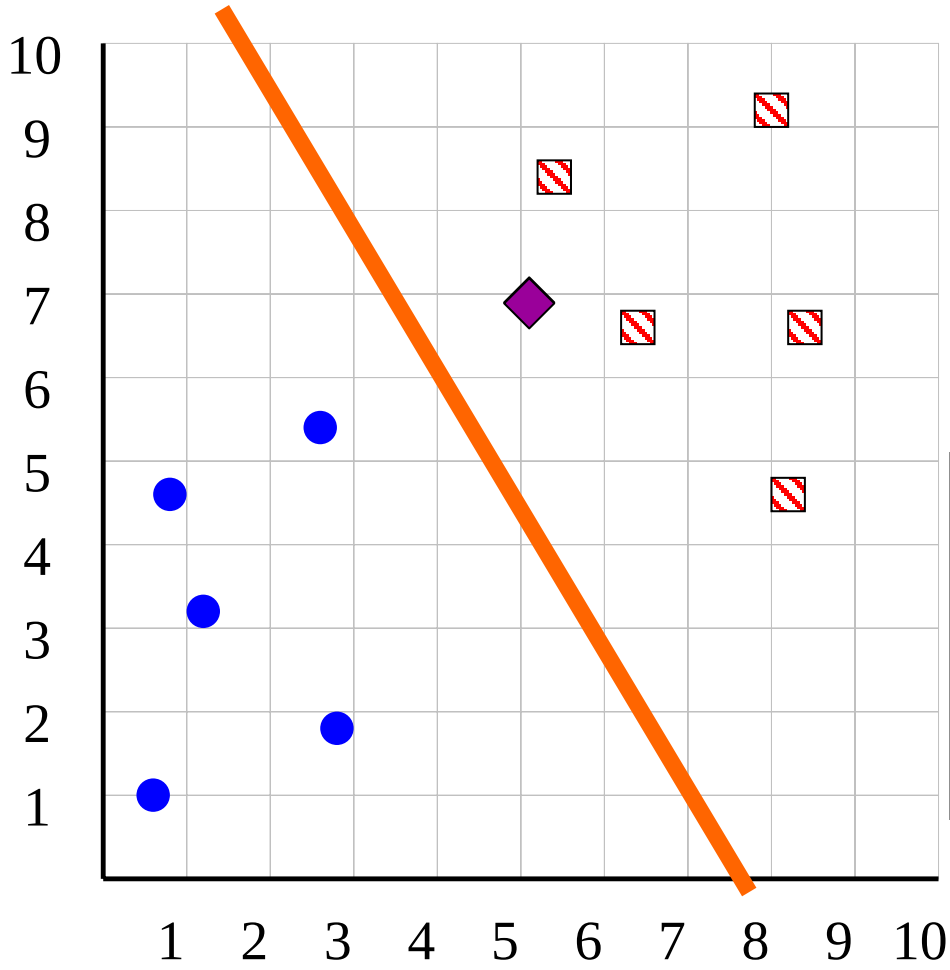
- Acabamos de abstrair os detalhes do nosso problema particular. Será muito mais fácil conversar sobre pontos no espaço.

▨ **Esperança**

• **Gafanhoto**

Classificador Linear Simples

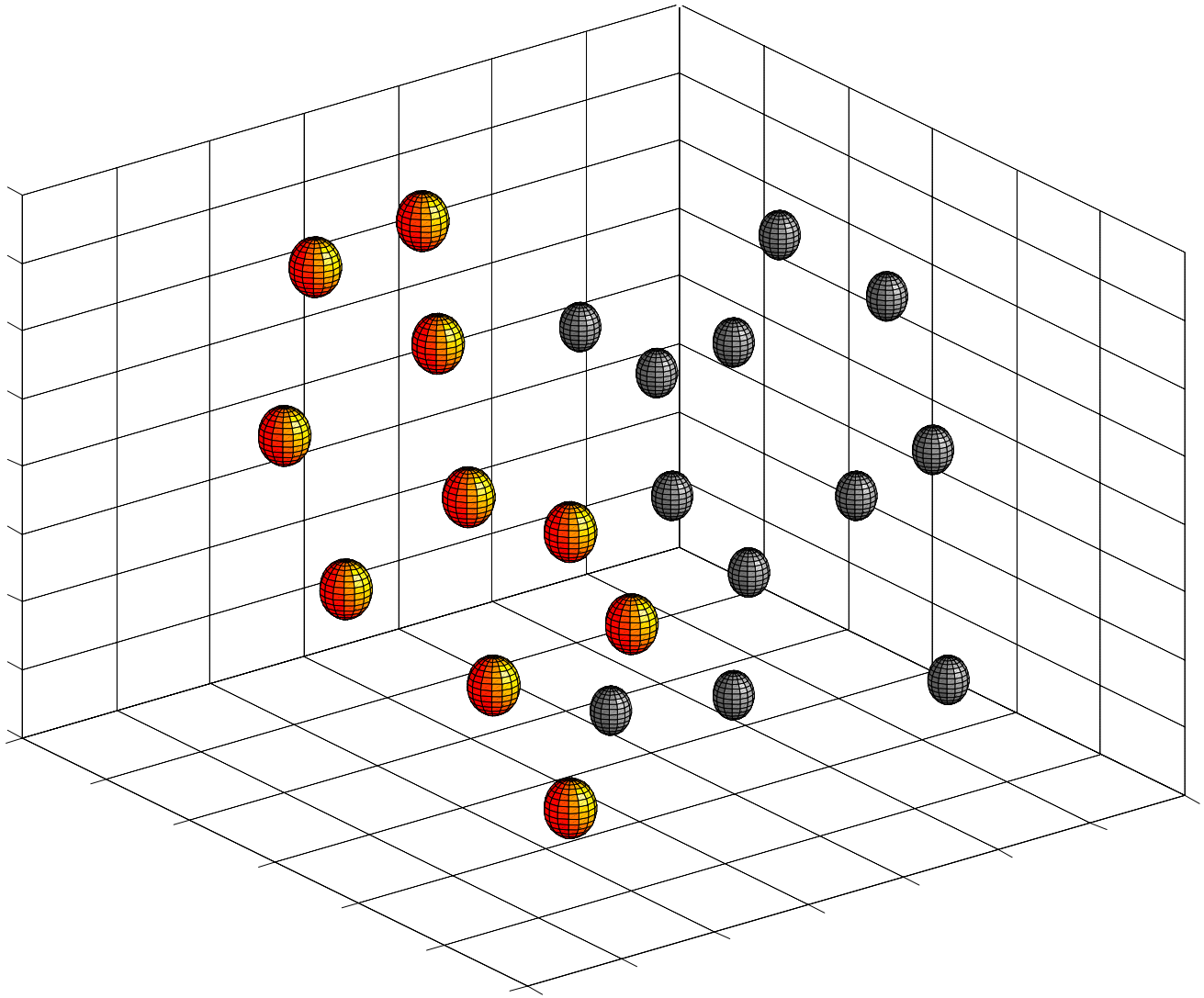
R.A. Fisher
1890-1962



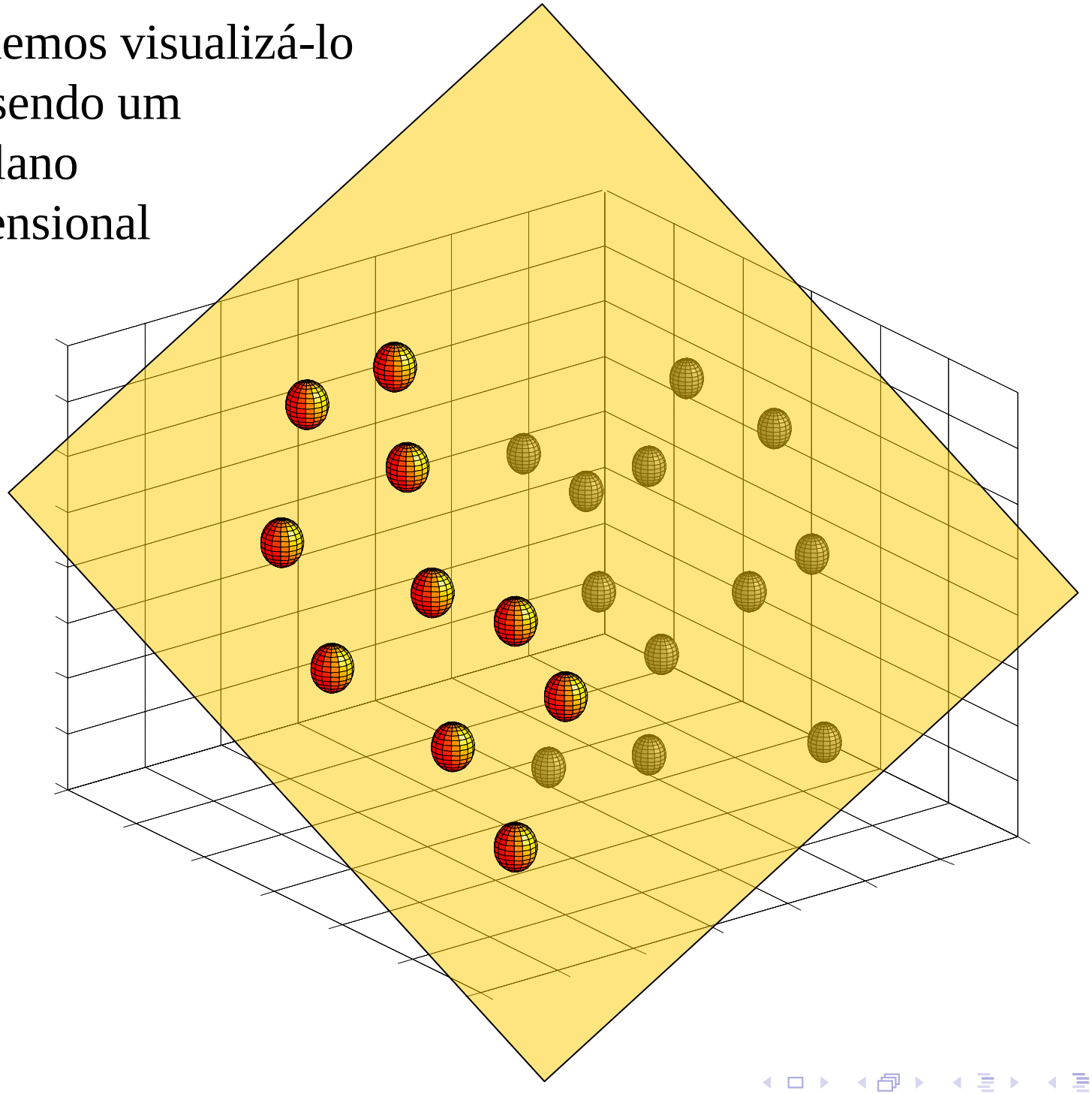
Se exemplo não visto antes está acima da linha
Então
classe é **Esperança**
senão
classe é **Gafanhoto**

Esperança
Gafanhoto

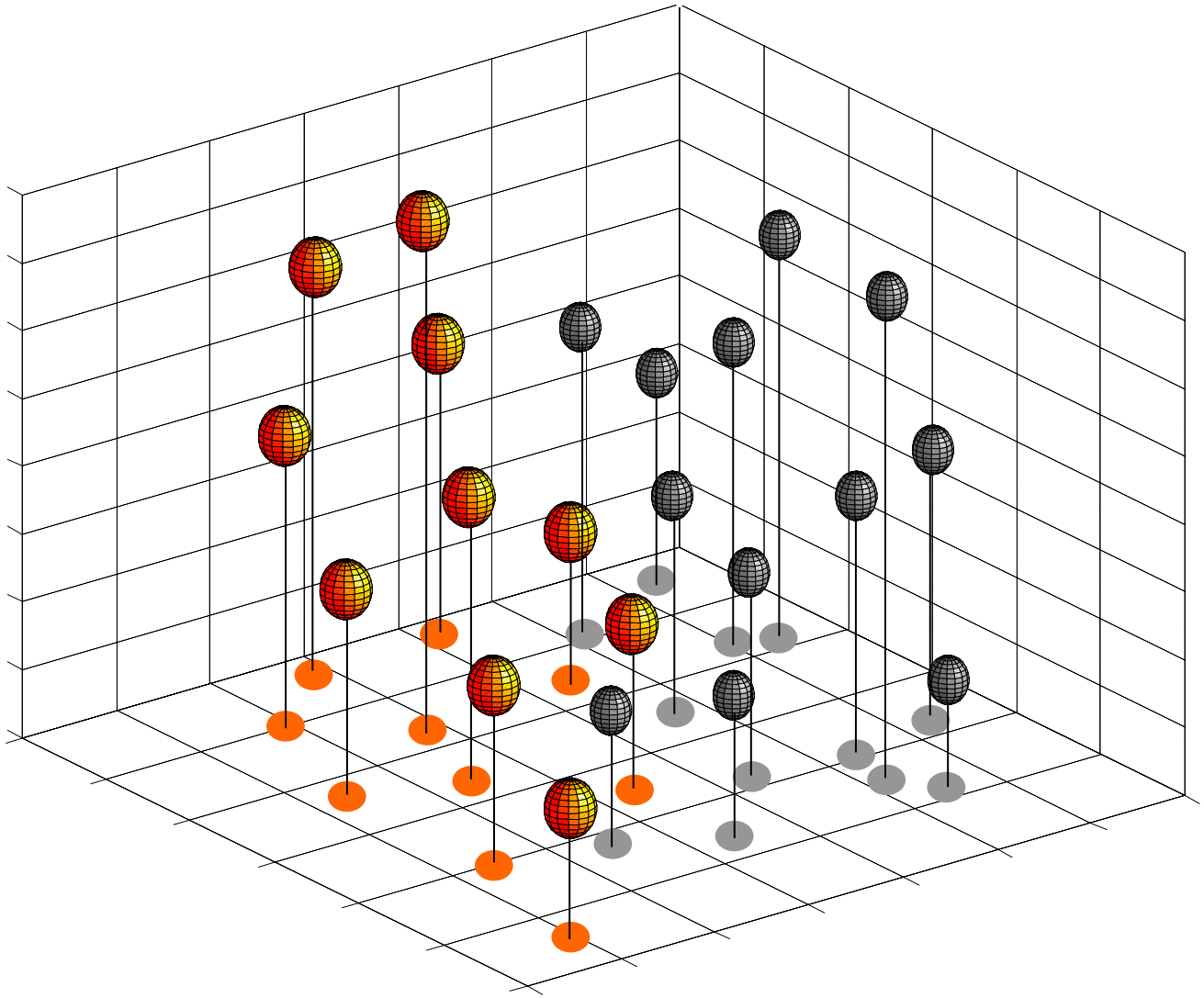
O classificador linear simples é definido para espaços dimensionais maiores…

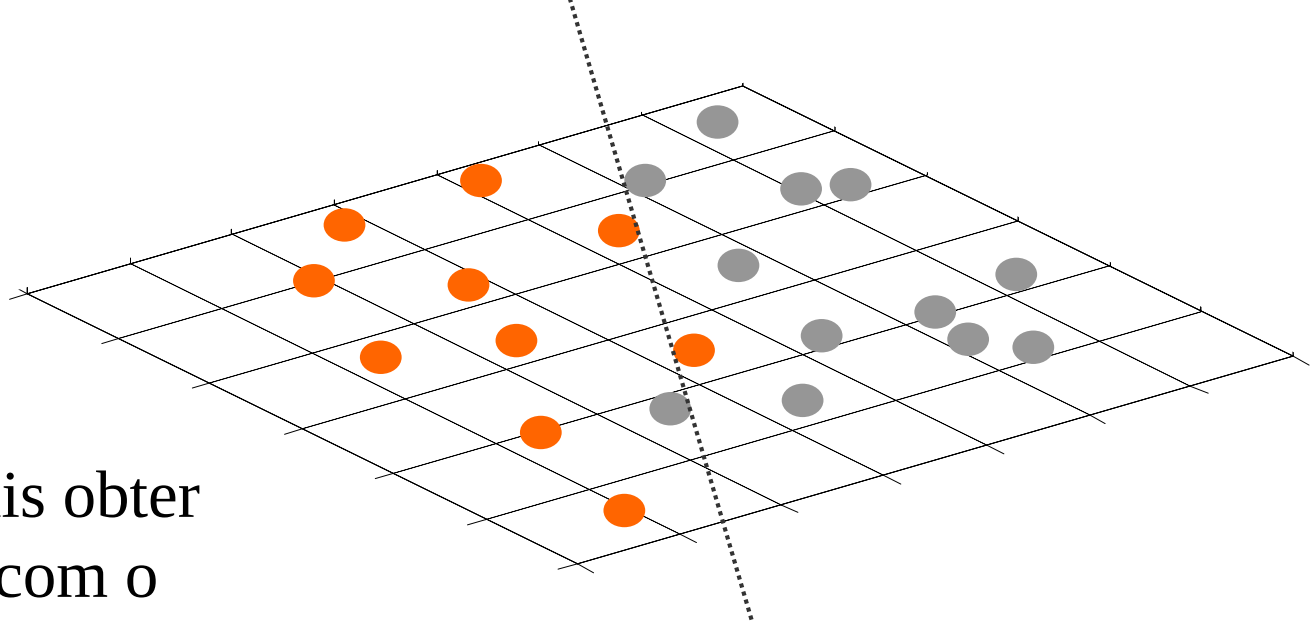


… podemos visualizá-lo como sendo um hiperplano n-dimensional



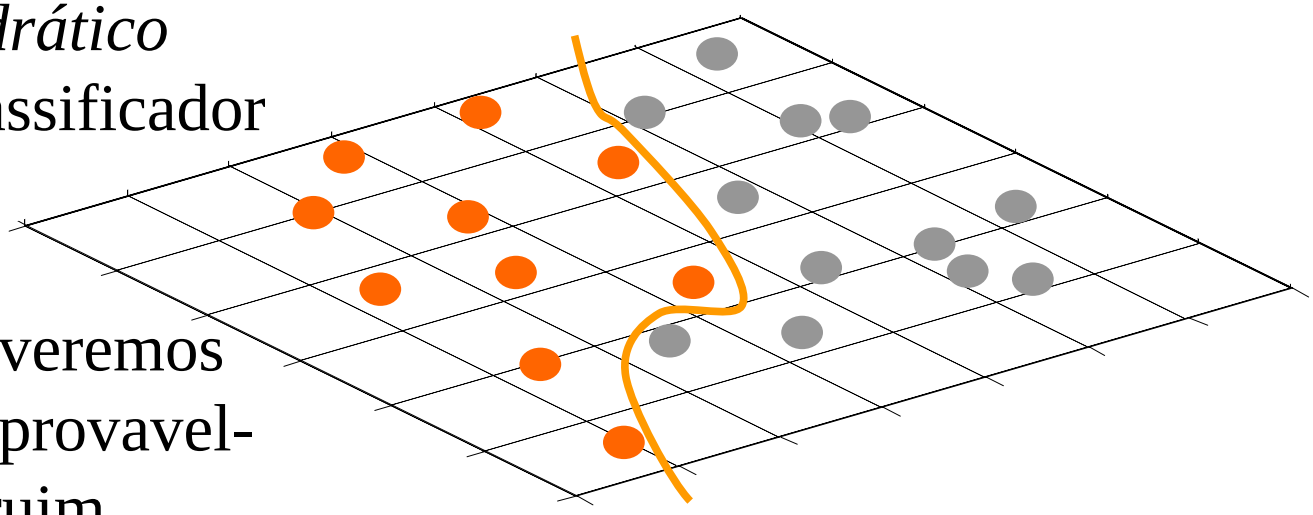
É interessante pensar no que aconteceria neste exemplo se não tivéssemos a terceira dimensão…





Não podemos mais obter acurácia perfeita com o classificador linear simples…

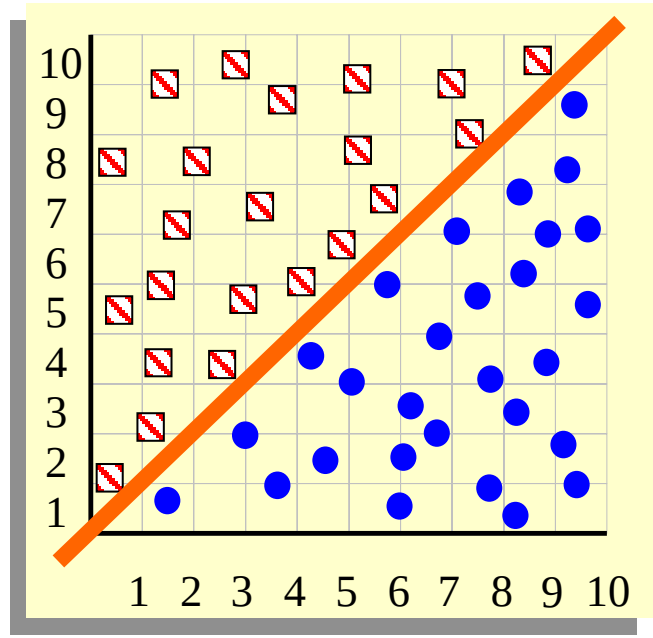
Podemos tentar resolver este problema usando um classificador *quadrático* simples ou um classificador *cúbico* simples…

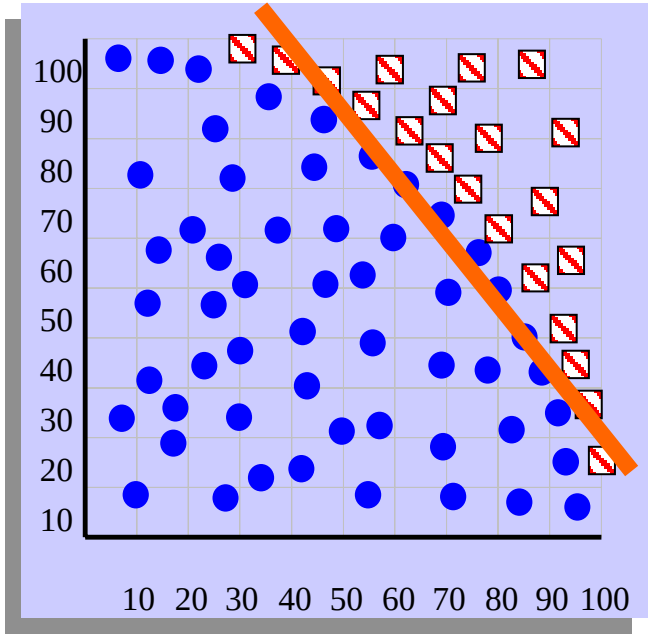


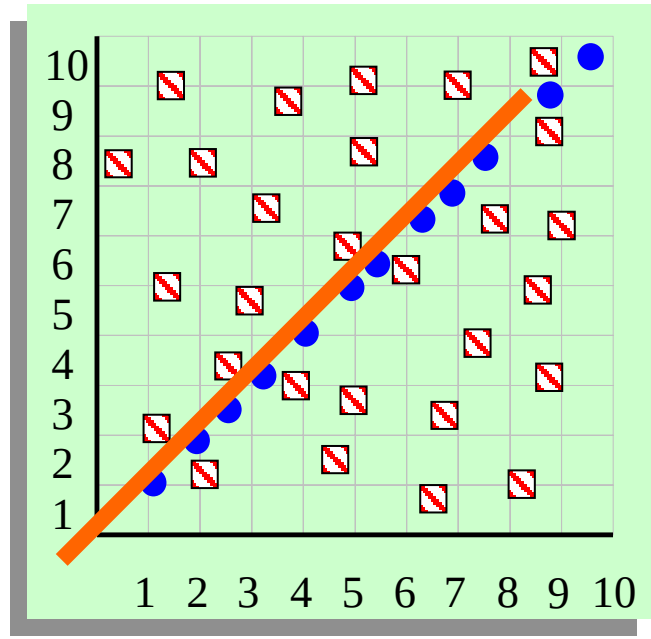
Entretanto, como veremos mais tarde, esta é provavelmente uma idéia ruim…

Quais dos “Problemas do Pombo” podem ser resolvidos pelo Classificador Linear Simples?

1) Perfeito
2) Inútil
3) Muito bom







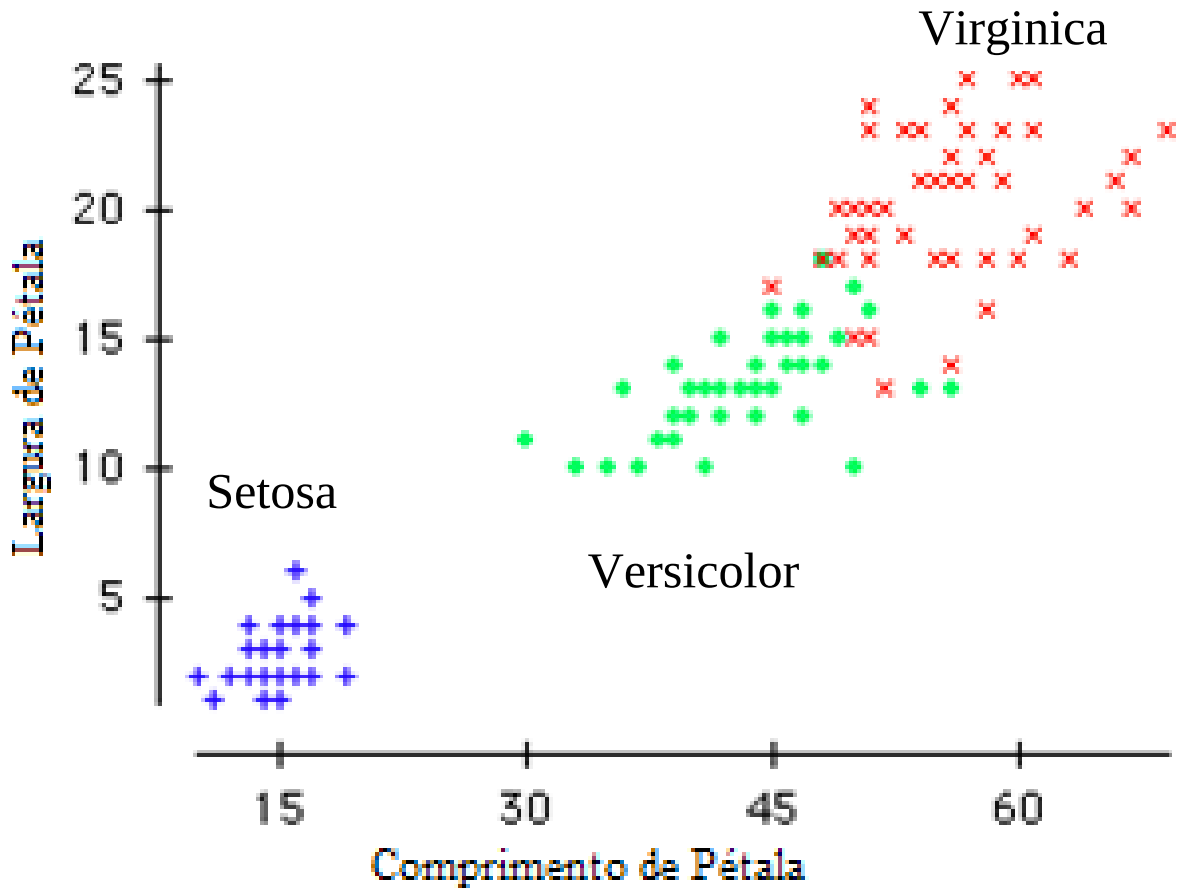
Problemas que podem ser resolvidos por um classificador linear são chamados de **linearmente separáveis**.

Um problema famoso

R. A. Fisher's Iris Dataset.

- 3 classes
- 50 exemplos de cada classe

A tarefa é classificar as plantas em uma das 3 variedades usando comprimento de pétala e largura de pétala.

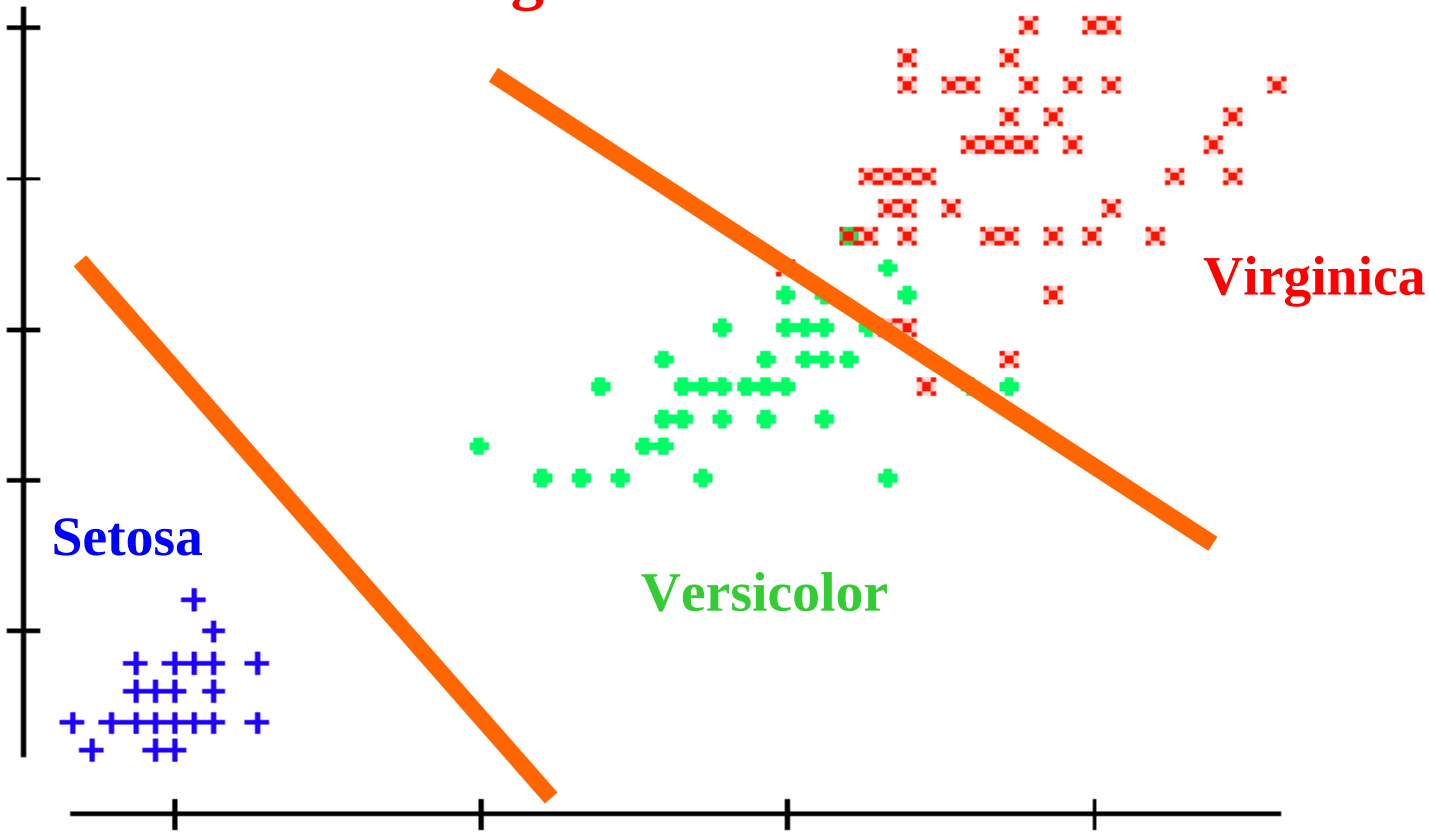


Iris Setosa

Iris Versicolor

Iris Virginica

Podemos generalizar o classificador linear relativo a variáveis a N classes, combinando N-1 linhas. Neste caso primeiramente aprendemos a linha para (perfeitamente) discriminar entre **Setosa** e **Virginica**/**Versicolor**, então aprendemos a discriminar aproximadamente entre **Virginica** **e** **Versicolor**.



Se comp. de pétala > 3.272 – (0.325 * comp. de pétala)
Então classe = **Virginica** **Senão Se** larguar de pétala…

Vimos agora um algoritmo de classificação e estamos prestes a ver mais. Como deveríamos compará-los?

- Acurácia de predição
- Velocidade e Escalabilidade
  - Tempo para construir o modelo
  - Tempo para usar o modelo
  - Eficiência com bases de dados armazenadas em discos
- Robustez
  - Com o tratamento de ruído, valores faltantes e características irrelevantes, streaming de dados
- Interpretabilidade:
  - Compreensão e percepção fornecidas pelo modelo

Acurácia da Predição (I)

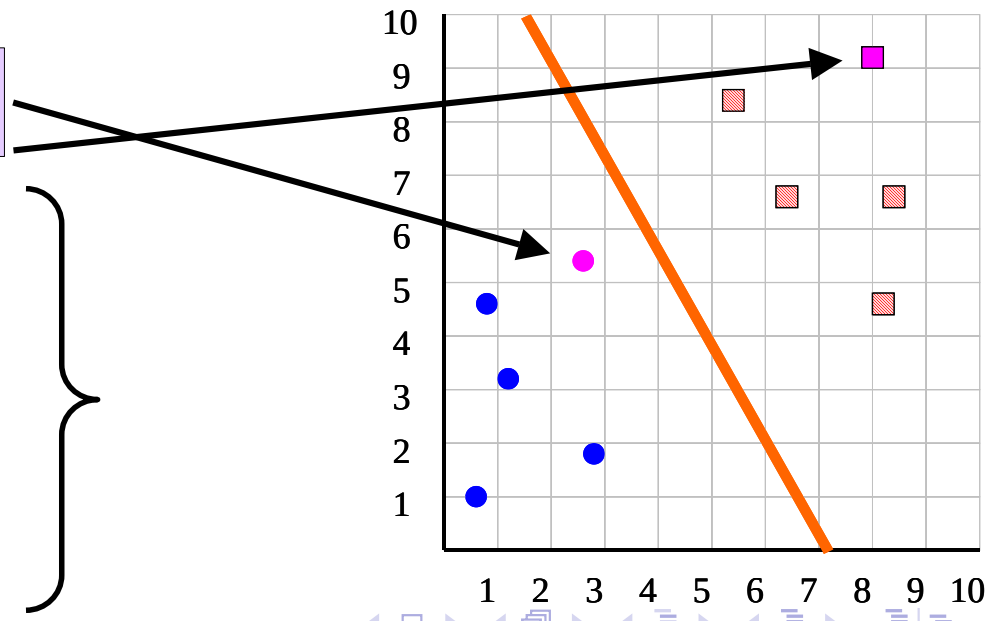
- Como *estimamos* a **acurácia** do nosso classificador?
  Podemos usar a **validação cruzada de *k*-folds**

Dividimos o conjunto de dados em *k* partes (subconjuntos) de tamanhos iguais. O algoritmo é testado *k* vezes e a cada iteração deixa-se uma das *k* partes de fora da construção do classificador, mas usa-se ela para *testar* o classificador

$$\text{Acurácia} = \frac{\text{Número de classificações corretas}}{\text{Número de exemplos em nossa base de dados}}$$

k = 5

ID do inseto	Comp. do abdomen	Comp. das antenas	Classe do Inseto
1	2.7	5.5	Gafanhoto
2	8.0	9.1	Esperança
3	0.9	4.7	Gafanhoto
4	1.1	3.1	Gafanhoto
5	5.4	8.5	Esperança
6	2.9	1.9	Gafanhoto
7	6.1	6.6	Esperança
8	0.5	1.0	Gafanhoto
9	8.3	6.6	Esperança
10	8.1	4.7	Esperança



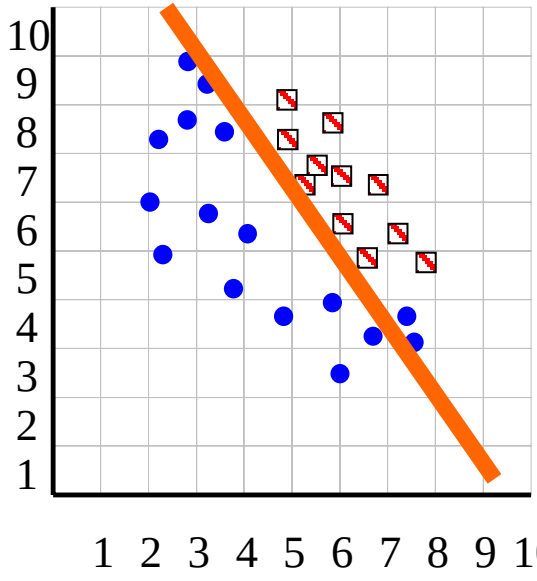
Acurácia de Predição (II)

• Usar a validação cruzada de *k*-folds é uma boa forma de estabelecer quaisquer parâmetros que possamos precisar ajustar no classificador.

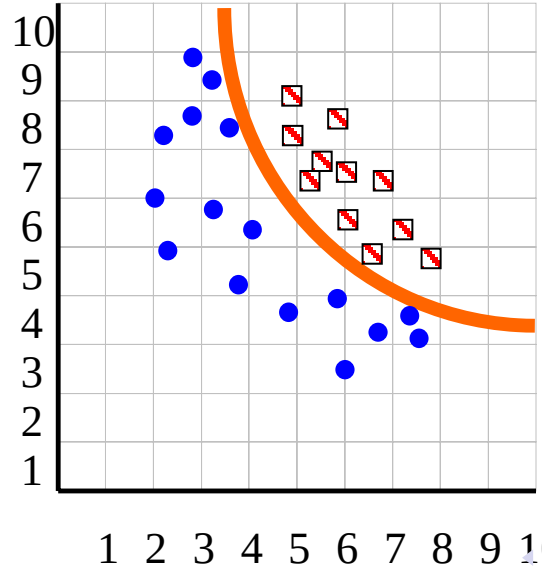
• Podemos fazer a validação cruzada de *k*-folds para qualquer conjunto possível e escolher o modelo com a maior acurácia. Onde houver um empate escolhemos o modelo mais simples.

• Na verdade, deveríamos provavelmente penalizar os modelos mais complexos, mesmo se eles tiverem maior acurácia, pois modelos mais complexos têm maior probabilidade de overfitting (discutido mais a frente).

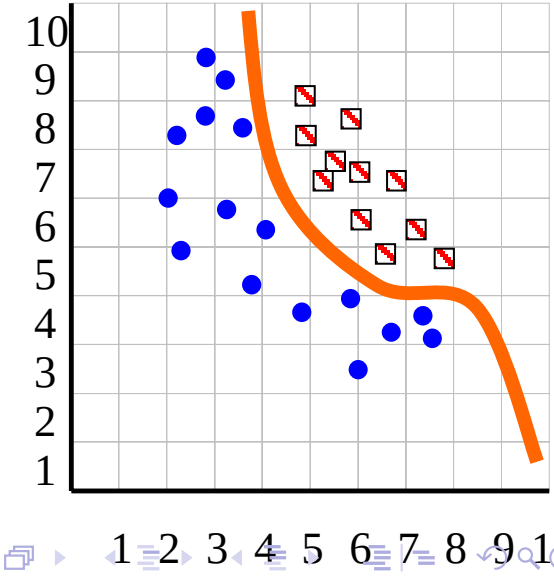
Acuidade = 94%



Acuidade = 100%



Acuidade = 100%



Acurácia de Predição (III)

$$\text{Acurácia} = \frac{\text{Número de classificações corretas}}{\text{Número de exemplos na base de dados}}$$

Acurácia é um número único; podemos entender melhor se olharmos em uma **matriz de confusão**. Isso nos dá informações adicionais úteis…

Classe verdadeira é...

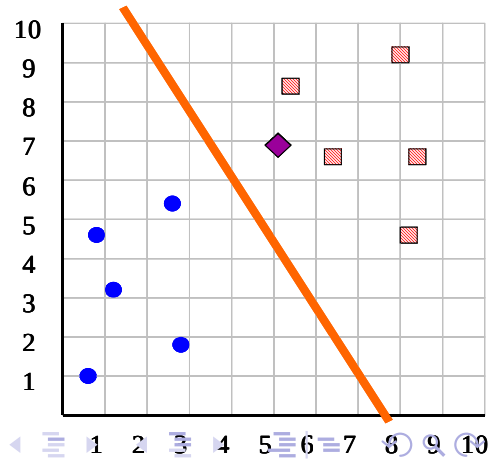
Classificado como um…	**Gato**	**Cão**	**Porco**
Gato	**100**	**0**	**0**
Cão	**9**	**90**	**1**
Porco	**45**	**45**	**10**

Velocidade e Escalabilidade I

Precisamos considerar as necessidades de tempo e de espaço para as duas fases distintas de classificação:

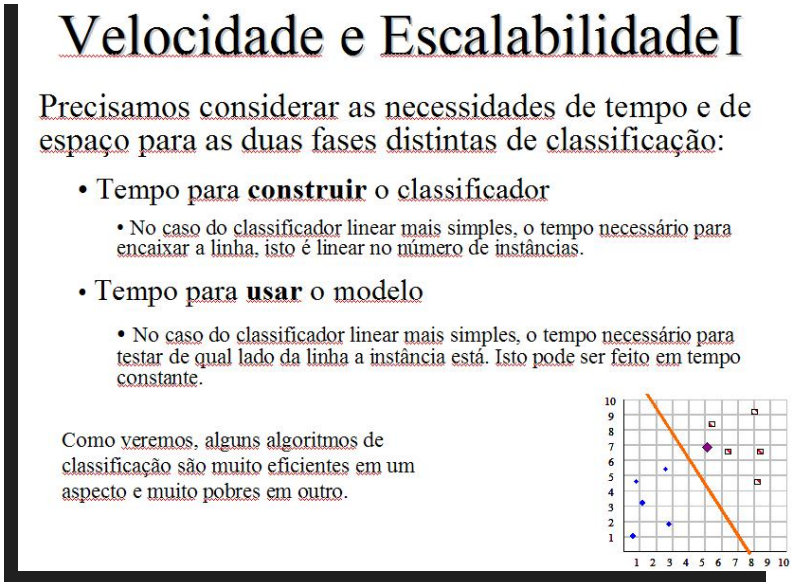
- Tempo para **construir** o classificador
  - No caso do classificador linear mais simples, o tempo necessário para ajustar a linha. Esse passo é linear no número de exemplos.
- Tempo para **usar** o modelo
  - No caso do classificador linear mais simples, o tempo necessário para testar de qual lado da linha o exemplo está. Isso pode ser feito em tempo constante.

Como veremos, alguns algoritmos de classificação são muito eficientes em um aspecto e muito pobres em outro.



Velocidade e Escalabilidade II

Para aprendizado com pequenas bases de dados, esta é a idéia geral



Porém, para mineração de conjuntos de dados massivos, não é a complexidade de tempo (da memória principal) que importa tanto e sim quantas vezes precisamos percorrer a base de dados.

Isto ocorre porque para a maioria das operações de mineração de dados, o tempo de acesso a disco domina completamente o tempo da CPU.

Para mineração de dados, os pesquisadores frequentemente relatam o número de vezes que você deve percorrer a base de dados.

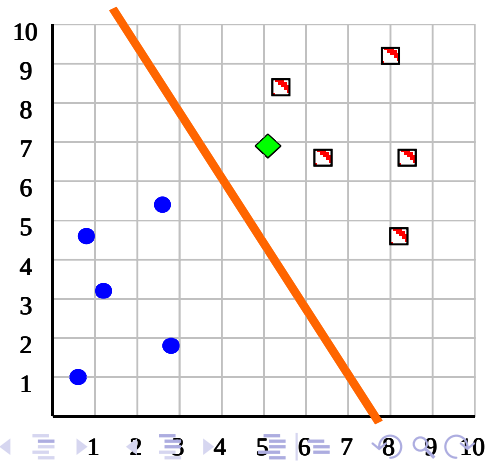
Robustez (I)

É preciso considerar o que acontece quando temos:

- Ruído
  - Por exemplo, a idade de uma pessoa pode ter sido digitada erroneamente como 650 ao invés de 65; como isto afeta nosso classificador? (Isto só é importante para construção do classificador, se o exemplo que queremos classificar tem ruído, não podemos fazer nada).

- Valores faltantes

  Por exemplo, suponha que queremos classificar um inseto, mas só conhecemos o comprimento do abdômen (eixo X), e não o comprimento das antenas (eixo Y); assim mesmo podemos classificar o exemplo?



Robustez (II)

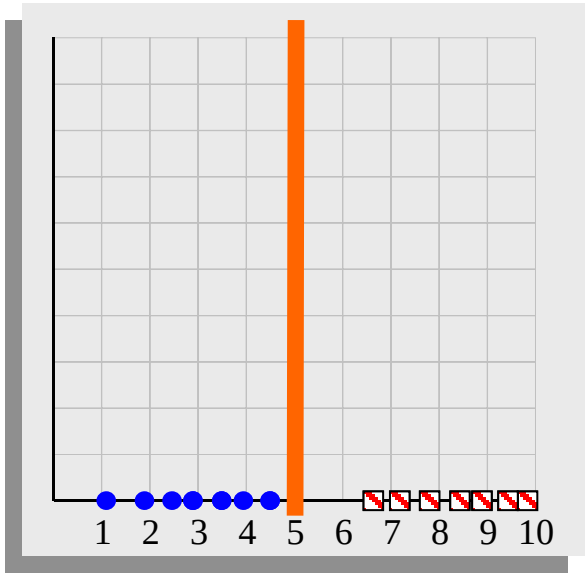
É preciso considerar o que acontece quando temos:

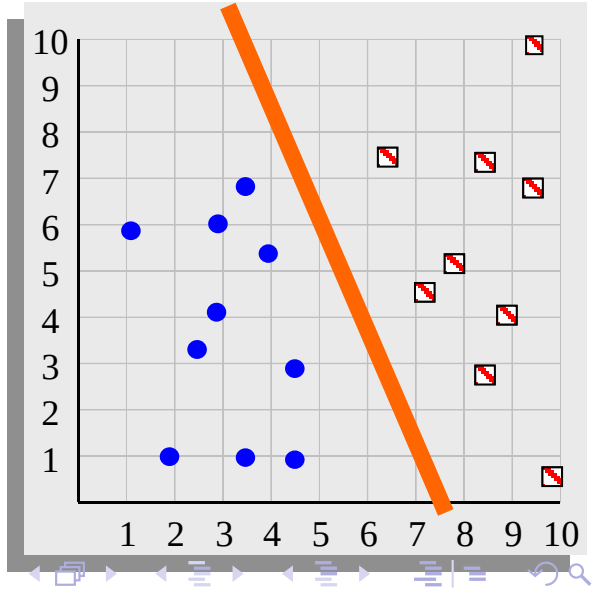
- Características irrelevantes

  Por exemplo, suponha que queremos classificar pessoas como
  - Aluno_Grad_Aprovado
  - Aluno_Grad_Nao_Aprovado

  E acontece que acertar mais que 5 em um teste em particular significa um indicador perfeito para o problema…

Se também usarmos "comprimento_cabelo" como uma característica, como isto afetará nosso classificador?





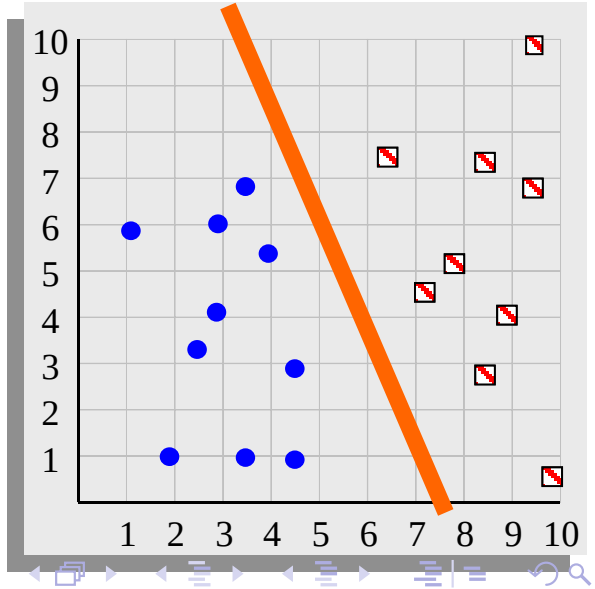
Robustez (III)

É preciso considerar o que acontece quando temos:

- Transmissão contínua de dados

  Para muitos problemas do mundo real, não temos um único conjunto de dados fixo. Ao contrário, o conjunto de dados chega constantemente, potencialmente para sempre… (mercado de valores, dados de previsão de tempo, dados de sensores, etc)

  Nosso classificador é capaz de lidar com transmissão contínua de dados?

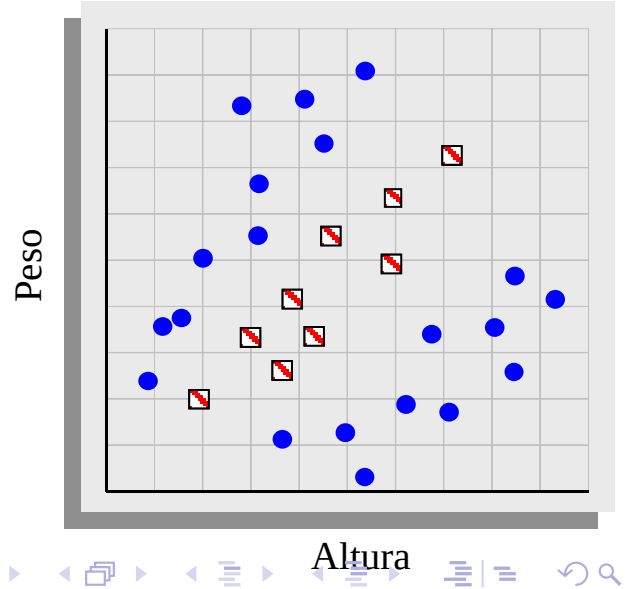


Interpretabilidade

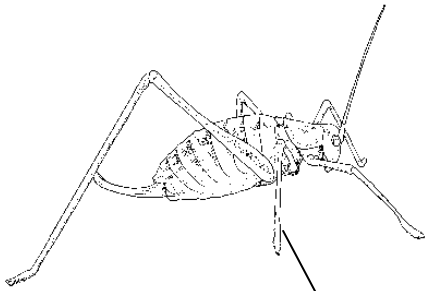
Alguns classificadores oferecem uma característica *bônus*. A estrutura do classificador aprendido nos diz algo sobre o domínio.

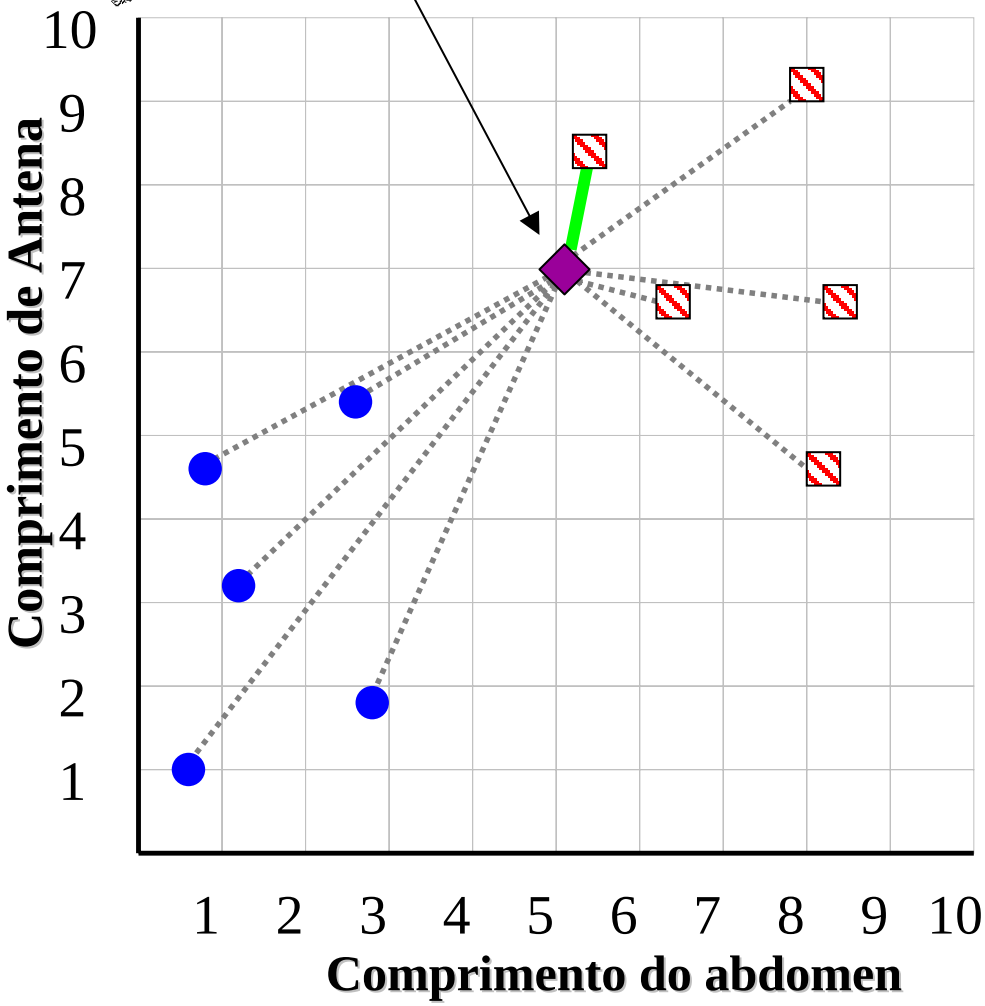
Como um exemplo trivial, se tentarmos classificar o risco de saúde de pessoas por apenas sua altura e peso, podemos ganhar a seguinte percepção (baseado na observação de que um único classificador linear não funciona bem, mas dois classificadores lineares funcionam).

Existem duas formas de não se estar saudável, estar obeso ou magro demais.



Classificador Vizinho Mais Próximo





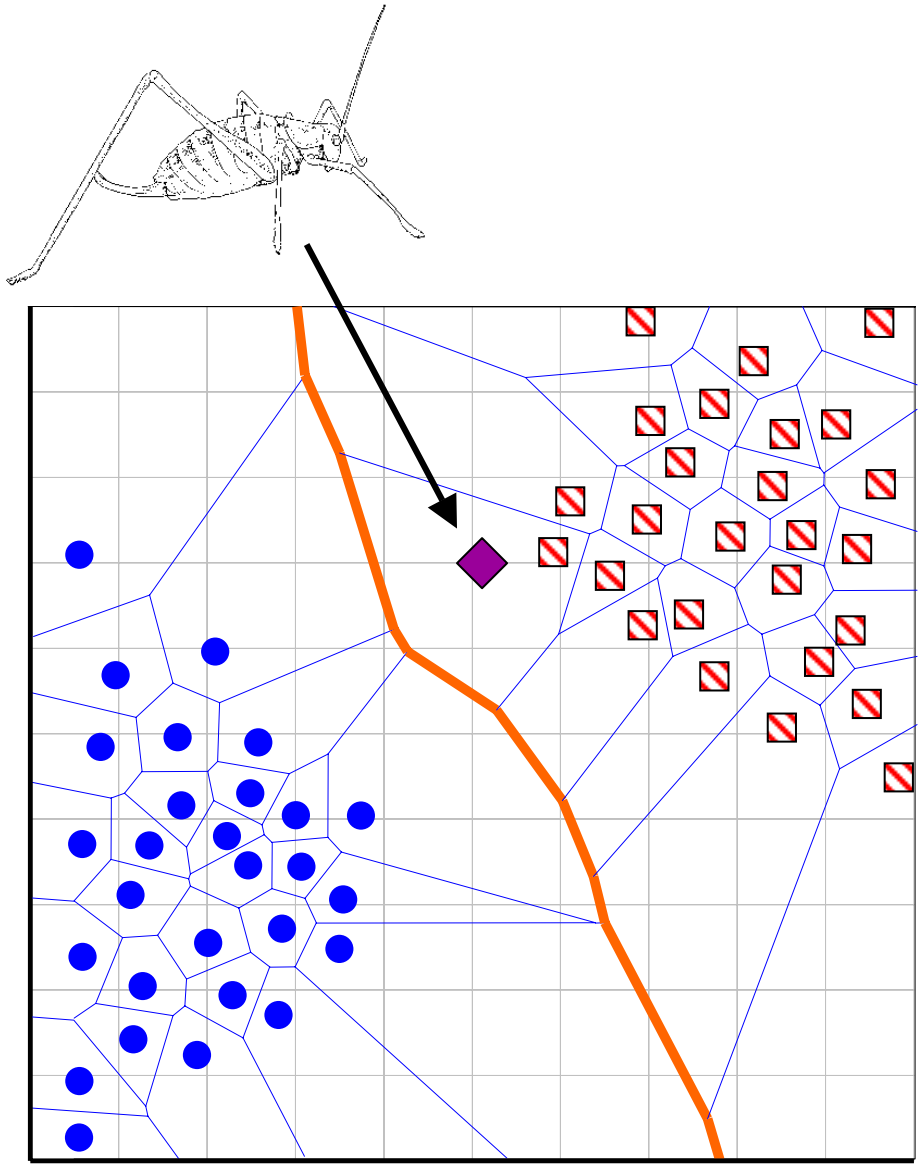
Evelyn Fix
1904-1965

Joe Hodges
1922-2000

Se o exemplo **mais próximo** de um exemplo não visto antes **é uma Esperança**
a classe é **Esperança**
Senão
a classe é **Gafanhoto**

Esperança
Gafanhotos

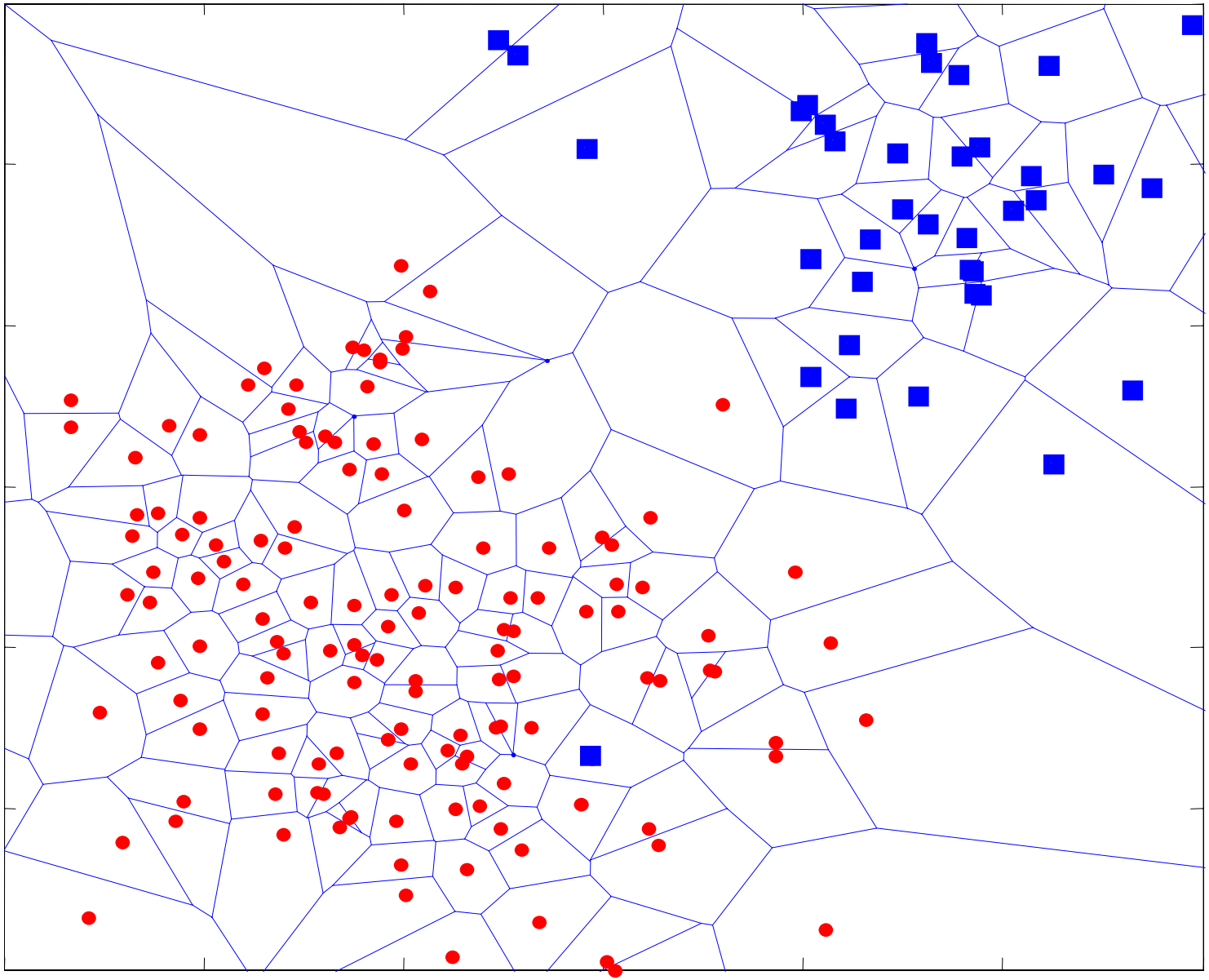
Podemos visualizar o algoritmo do vizinho mais próximo em termos de uma superfície de decisão…



Note que não precisamos realmente construir essas superfícies, elas são simplesmente os limites implícitos que dividem o espaço em regiões que "pertencem" a cada exemplo.

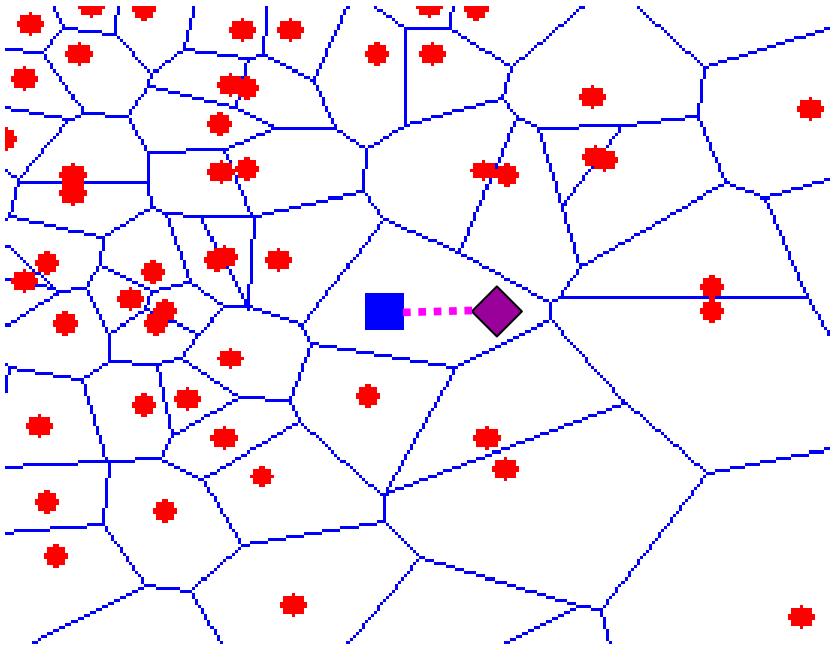
Esta divisão de espaço é chamada de Dirichlet Tessellation (ou diagrama de Voronoi, ou regiões Theissen).

O alg. do vizinho mais próximo é sensível a “exceções”…

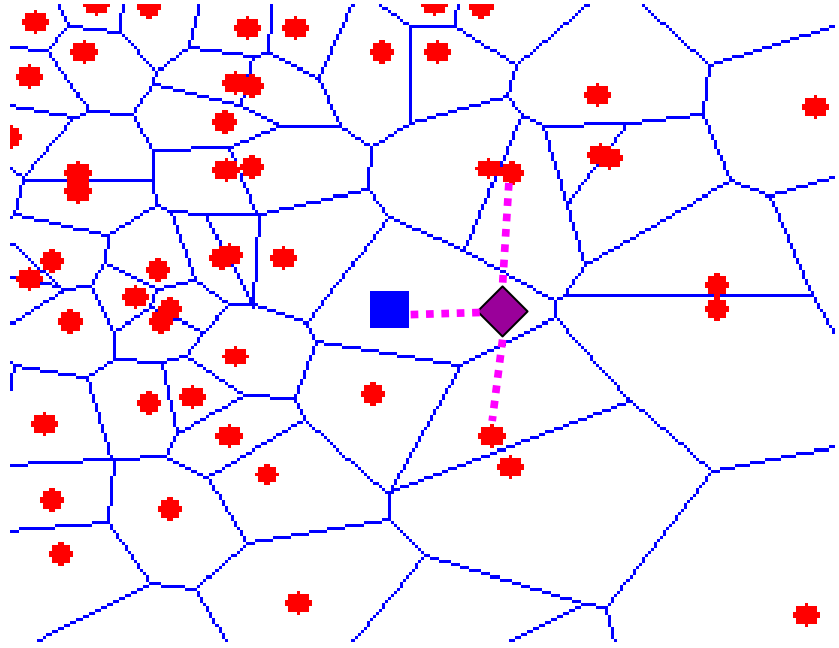


A solução é…

Podemos generalizar o algoritmo do vizinho mais próximo para o algoritmo do k-vizinhos mais próximos (KNN). Medimos a distância até os *k* exemplos mais próximos e as deixamos votar. *k* é tipicamente escolhido como um número ímpar.



$k = 1$



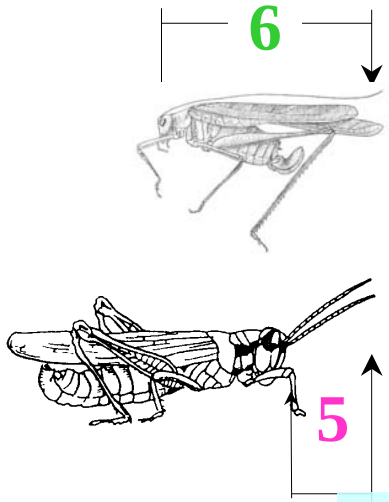
$k = 3$

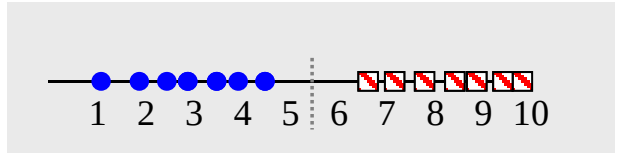
O algoritmo do vizinho mais próximo é sensível a características irrelevantes…

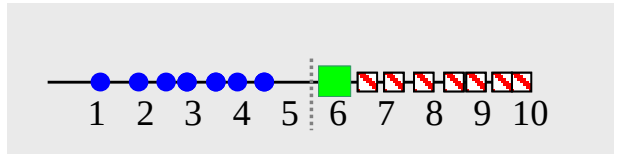
Suponha que o seguinte é verdadeiro, se a antena de um inseto é maior que 5.5 ele é um **Esperança**, senão ele é um **Gafanhoto**.

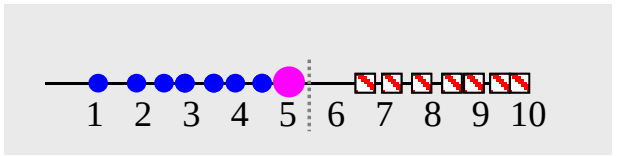
Usando somente o comprimento de antena conseguimos classificação perfeita!

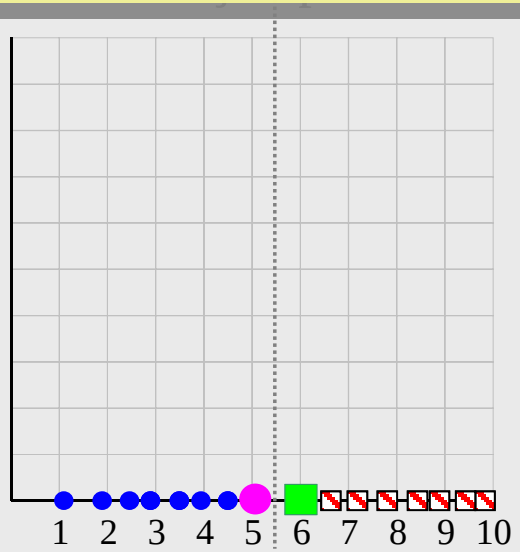
Dados de treinamento

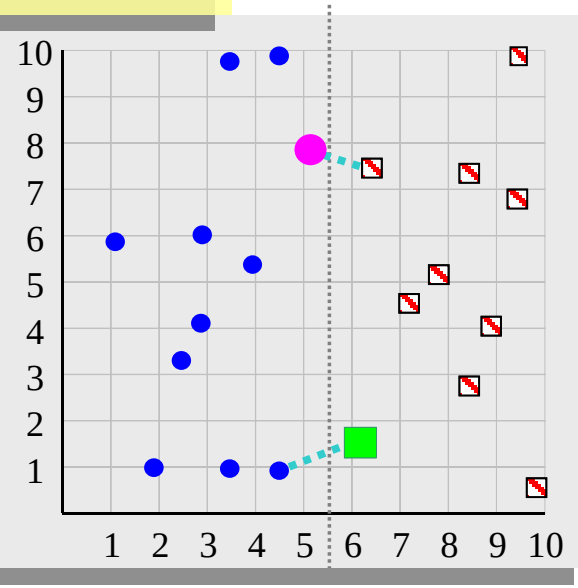












Suponha entretanto, que adicionemos uma característica **irrelevante**, por exemplo, a massa de um inseto.

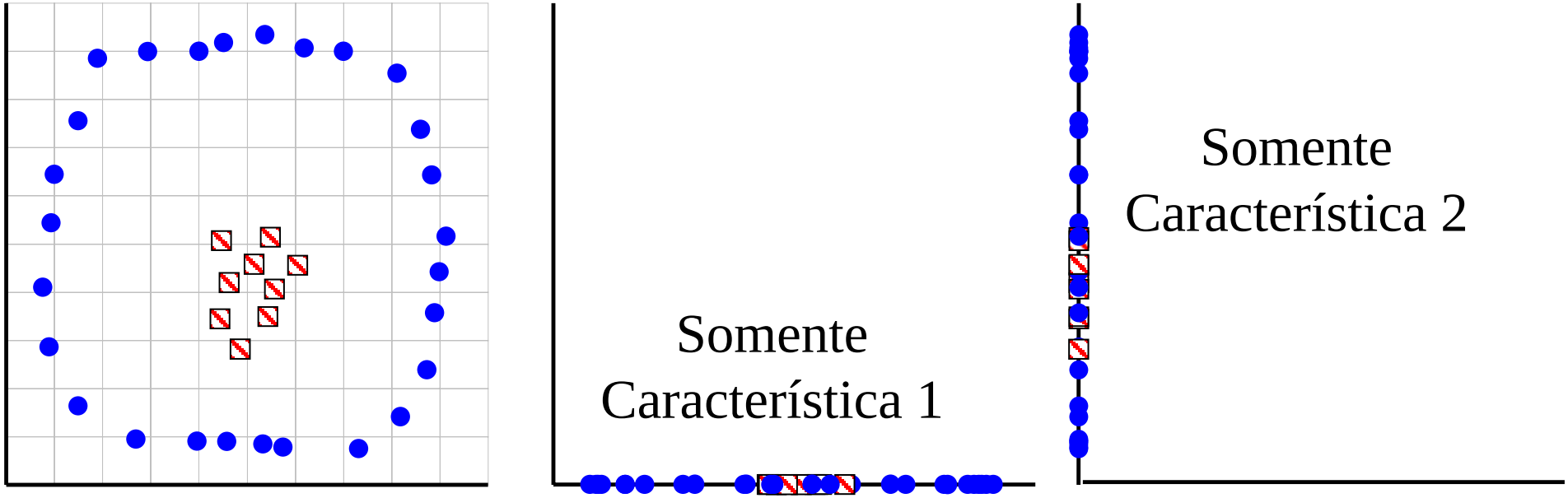
Usando o comprimento da antena e a massa dos insetos com o algoritmo 1-NN obtemos a classificação errada!

Como amenizamos a sensibilidade dos algoritmos do vizinho mais próximo a características irrelevantes?

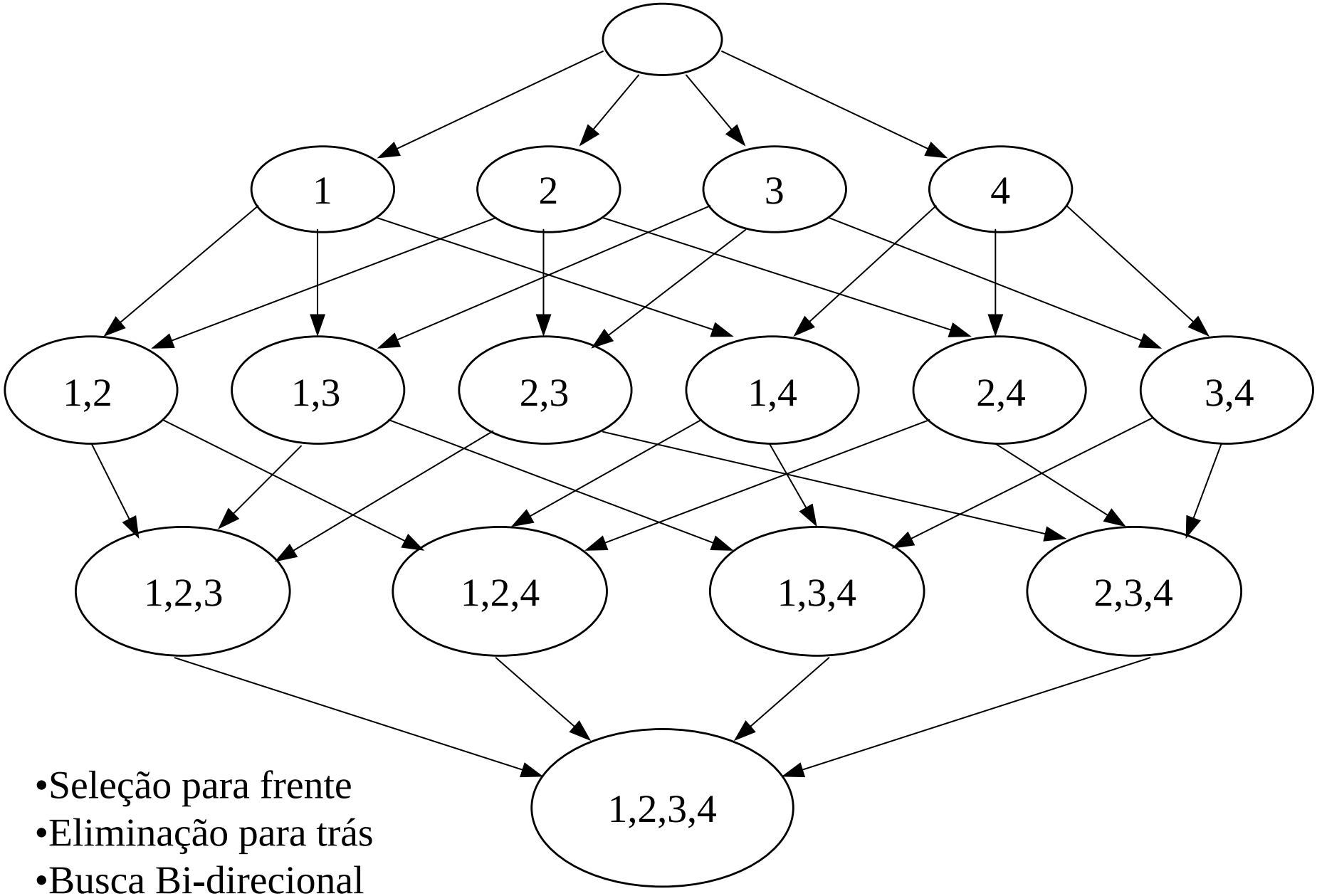
- Usando mais exemplos de treinamento
- Perguntando a um especialista quais características são relevantes para a tarefa
- Usando testes estatísticos para tentar determinar quais características são úteis
- Procurando sub-conjuntos de características (no próximo slide veremos porque isto é difícil)

Por que procurar sub-conjuntos de características é difícil

Suponha que você tenha o seguinte problema de classificação, com 100 características, e aconteça que as Características 1 e 2 (o X e Y abaixo) dão classificação perfeita, mas todas as outras 98 características são irrelevantes…

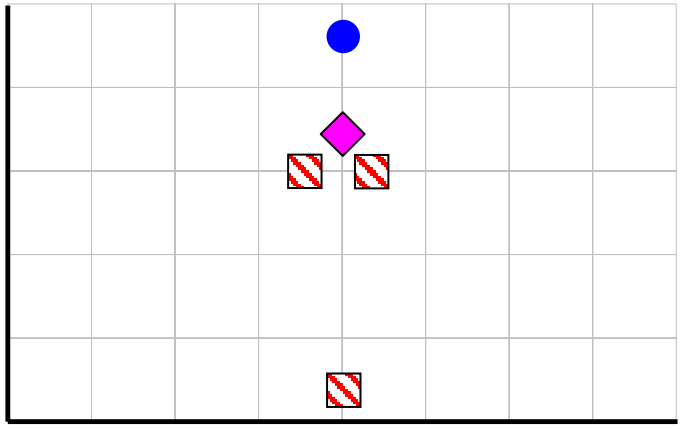


Usar todas as 100 características dará resultados pobres, mas também dará se usarmos somente a Característica 1, e também usando somente a Característica 2! Dos 2^{100} –1 possíveis sub-conjuntos de características, somente um realmente funcionará.



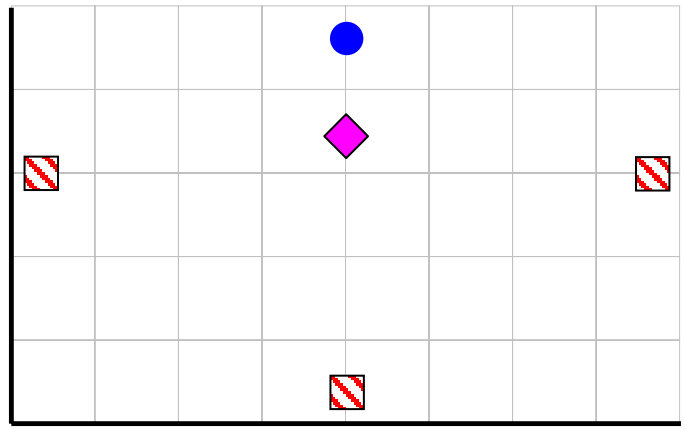
- Seleção para frente
- Eliminação para trás
- Busca Bi-direcional

O algoritmo do vizinho mais próximo é sensivel a unidades de medida



Eixo X medido em **centímetros**
Eixo Y medido em dólares

O vizinho mais próximo ao exemplo **cor-de-rosa** desconhecido é **vermelha**.



Eixo X medido em **milímetros**
Eixo Y medido em dólares

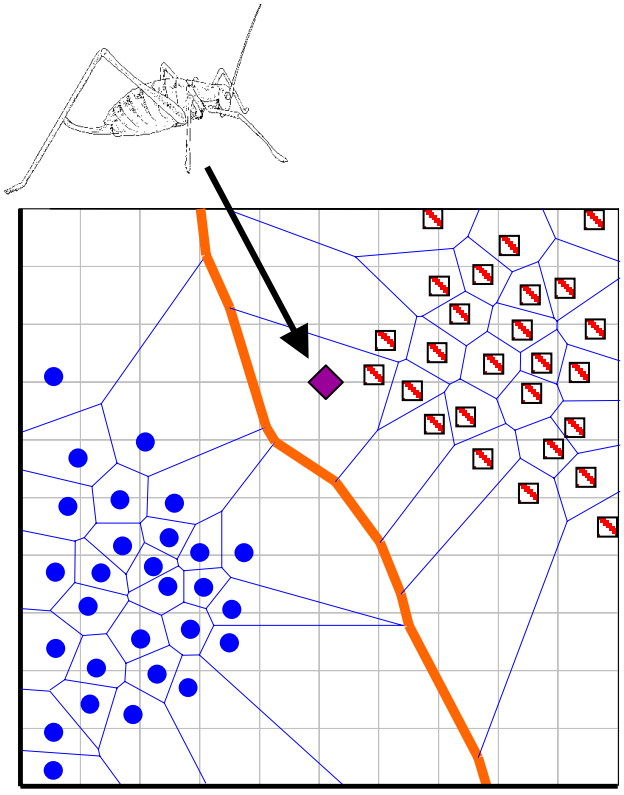
O vizinho mais próximo ao exemplo **cor-de-rosa** desconhecido é **azul**.

Uma solução é normalizar as unidades para números puros. Tipicamente as características são Z-normalizadas para ter uma média de zero e um desvio padrão de um. $X = (X - \text{mean}(X))/\text{std}(x)$

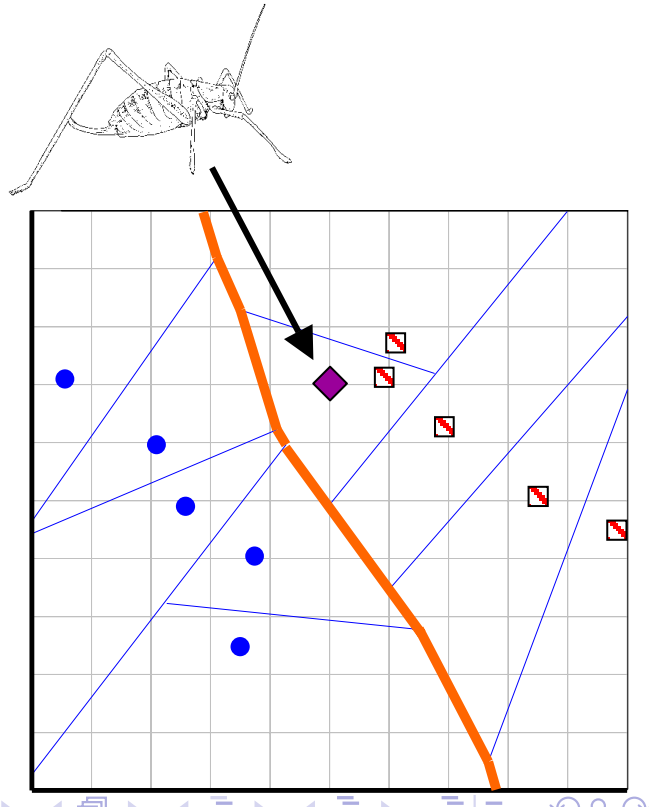
Podemos acelerar o algoritmo do vizinho mais próximo "jogando fora" alguns dados. Isto é chamado de limpeza de dados.

Note que isto pode as vezes melhorar a acurácia!

Também podemos acelerar a classificação com indexação

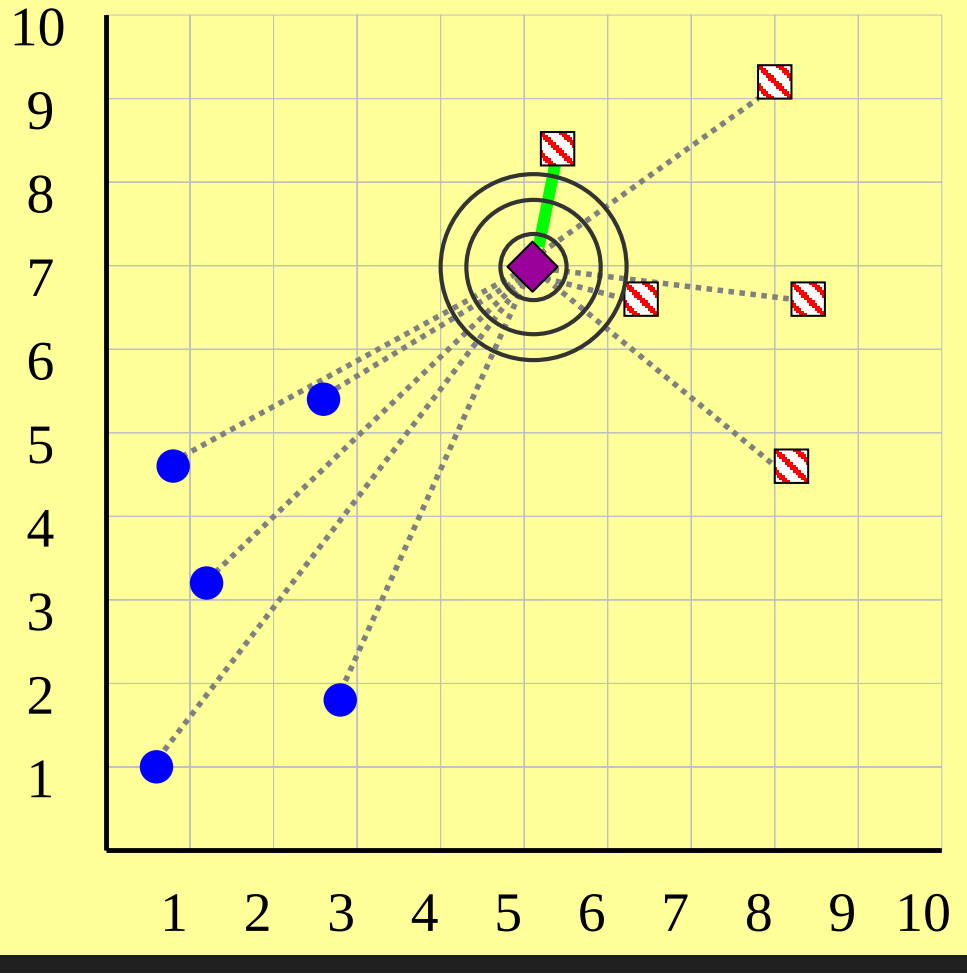


Uma abordagem possível. Apagar todos os exemplos que estão rodeados por membros das suas próprias classes.



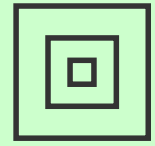
Até agora assumimos que o algoritmo do vizinho mais próximo usa a Distância Euclidiana, entretanto, este pode não ser o caso…

$$D(Q,C) \equiv \sqrt{\sum_{i=1}^{n}(q_i - c_i)^2}$$

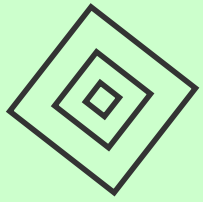


$$D(Q,C) \equiv \sqrt[p]{\sum_{i=1}^{n}(q_i - c_i)^p}$$

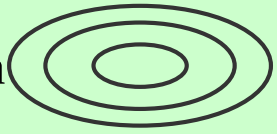
Max (p=inf)



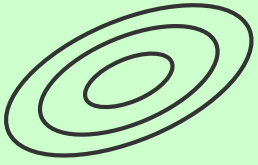
Manhattan (p=1)



Euclidiana Balanceada



Mahalanobis



…De fato, podemos usar o algoritmo do vizinho mais próximo com quaisquer funções de distância/similaridade

Por exemplo, "*Faloutsos*" é grego ou irlandês? Podemos comparar o nome "*Faloutsos*" com uma base de dados de nomes usando a distância de edição de seqüências de caracteres…

editar_distância (*Faloutsos*, *Keogh*) = 8
editar_distância (*Faloutsos*, *Gunopulos*) = 6

Com sorte, a semelhança do nome (particularmente o sufixo) com outros nomes gregos pode significar que o vizinho mais próximo é também um nome grego.

ID	Name	Classe
1	Gunopul**os**	**Grego**
2	Papadopoul**os**	**Grego**
3	Kolli**os**	**Grego**
4	Dardan**os**	**Grego**
5	Keo**gh**	**Irlandês**
6	Gou**gh**	**Irlandês**
7	Greenhau**gh**	**Irlandês**
8	Hadlei**gh**	**Irlandês**

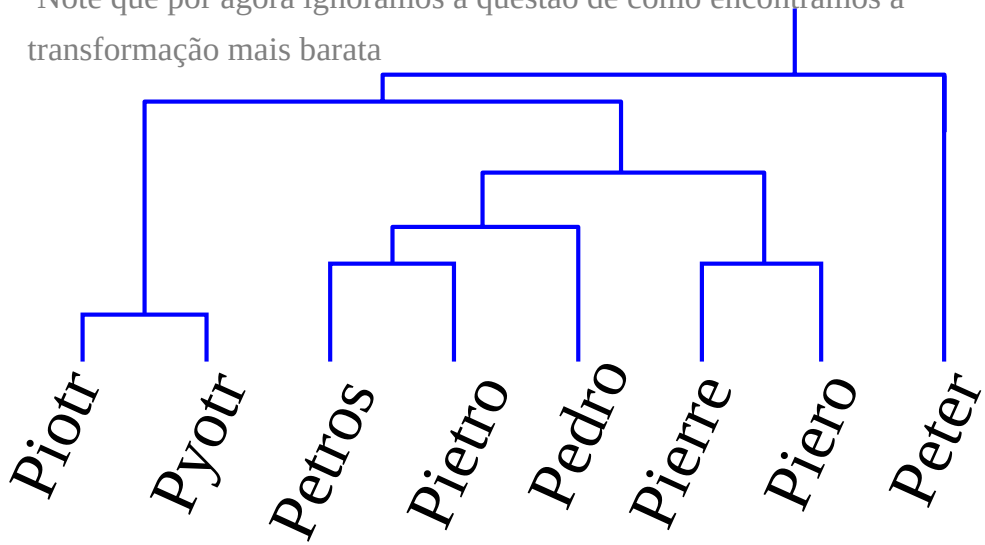
Medidas de distância especializadas existem para seqüências de DNA, séries temporais, imagens, grafos, vídeos, conjuntos, impressões digitais, etc…

Exemplo de Distância de Edição

É possível transformar qualquer string *Q* em uma string *C*, usando somente *Substituição*, *Inserção* e *Deleção*. Assuma que cada um destes operadores tem um custo associado.

A similaridade entre duas strings pode ser definida como o custo da transformação mais barata de *Q* para *C*.

Note que por agora ignoramos a questão de como encontramos a transformação mais barata

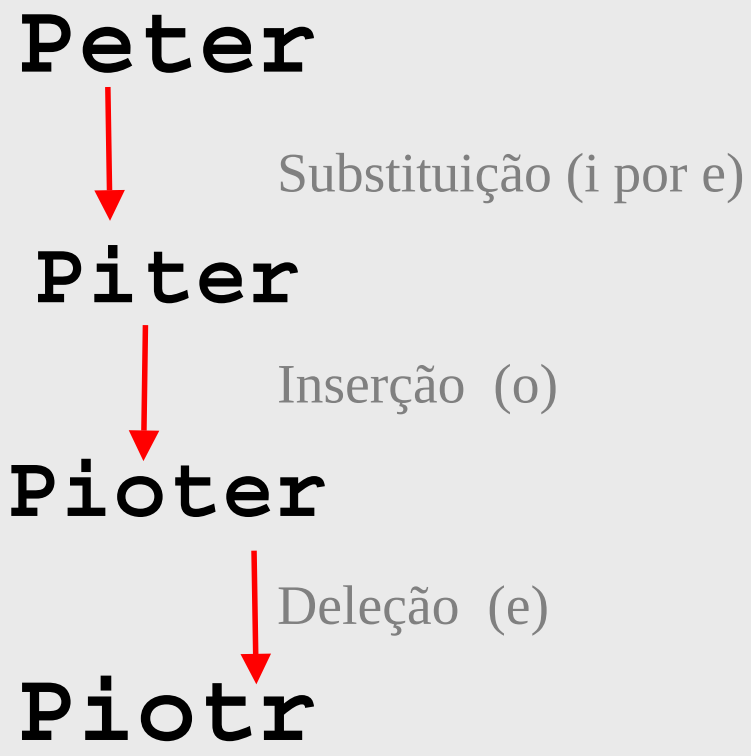


Quão semelhantes são os nomes “Peter” e “Piotr”?

Assuma a seguinte função de custo

Substituição	1 Unidade
Inserção	1 Unidade
Deleção	1 Unidade

D(`Peter,Piotr`) é 3



Referências I

[1] Eamonn Keogh,
Professor, Computer Science & Engineering Department, University of California - Riverside.
http://www.cs.ucr.edu/~eamonn/tutorials.html

[2] Monard, M. C.
Slides da disciplina SCC630 - Inteligência Artificial. ICMC - USP, 2010.