Aula 08 – Máquinas de vetores de suporte

1001524 – Aprendizado de Máquina I
2023/1 - Turmas A, B e C
Prof. Dr. Murilo Naldi

naldi@ufscar.br

Agradecimentos

- Parte do material utilizado nesta aula foi cedido pelos professores Diego Furtado Silva, Ricardo Cerri e, por esse motivo, o crédito deste material é deles
- Parte do material utilizado nesta aula foi disponibilizado por M. Kumar no endereço:
  - www-users.cs.umn.edu/~kumar/dmbook/index.php
- Agradecimentos a Intel Software e a

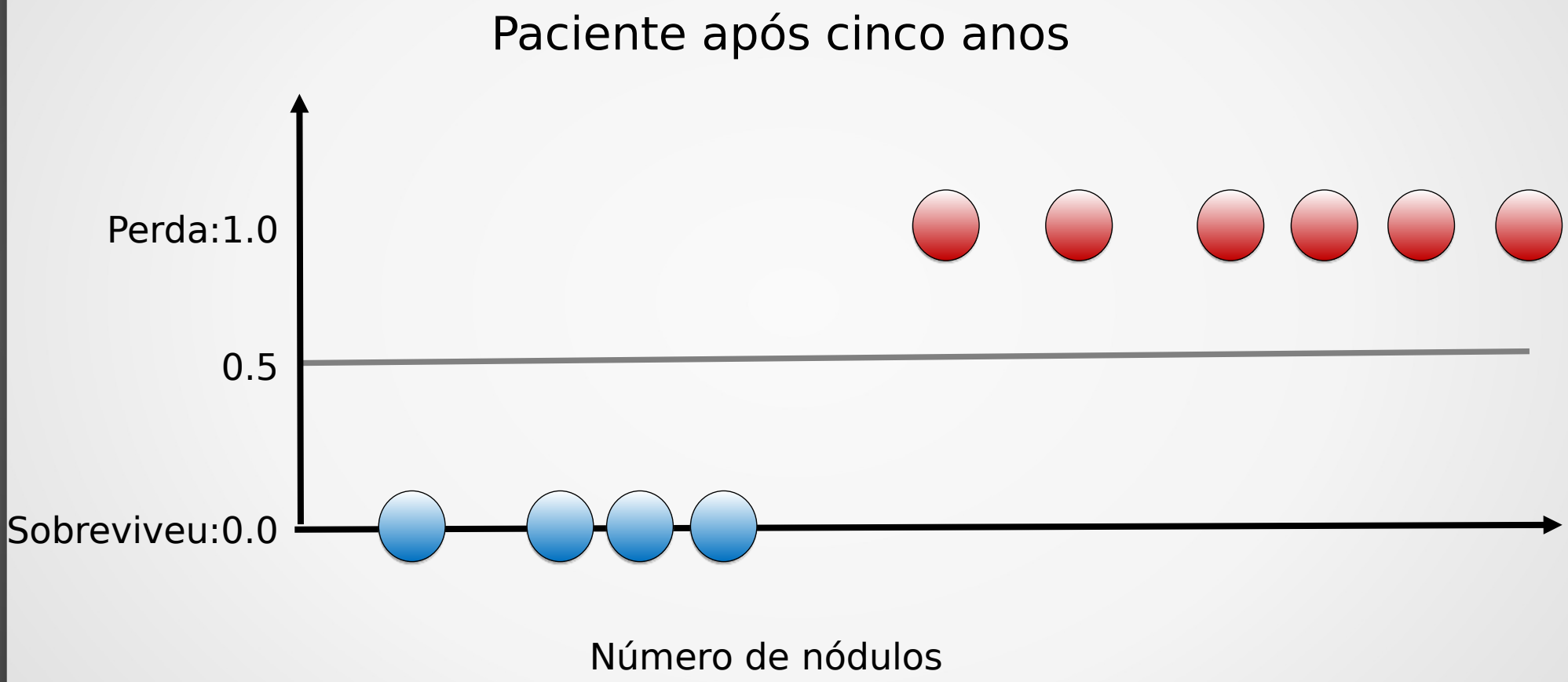
  Intel IA Academy pelo material

  disponibilizado e recursos didáticos

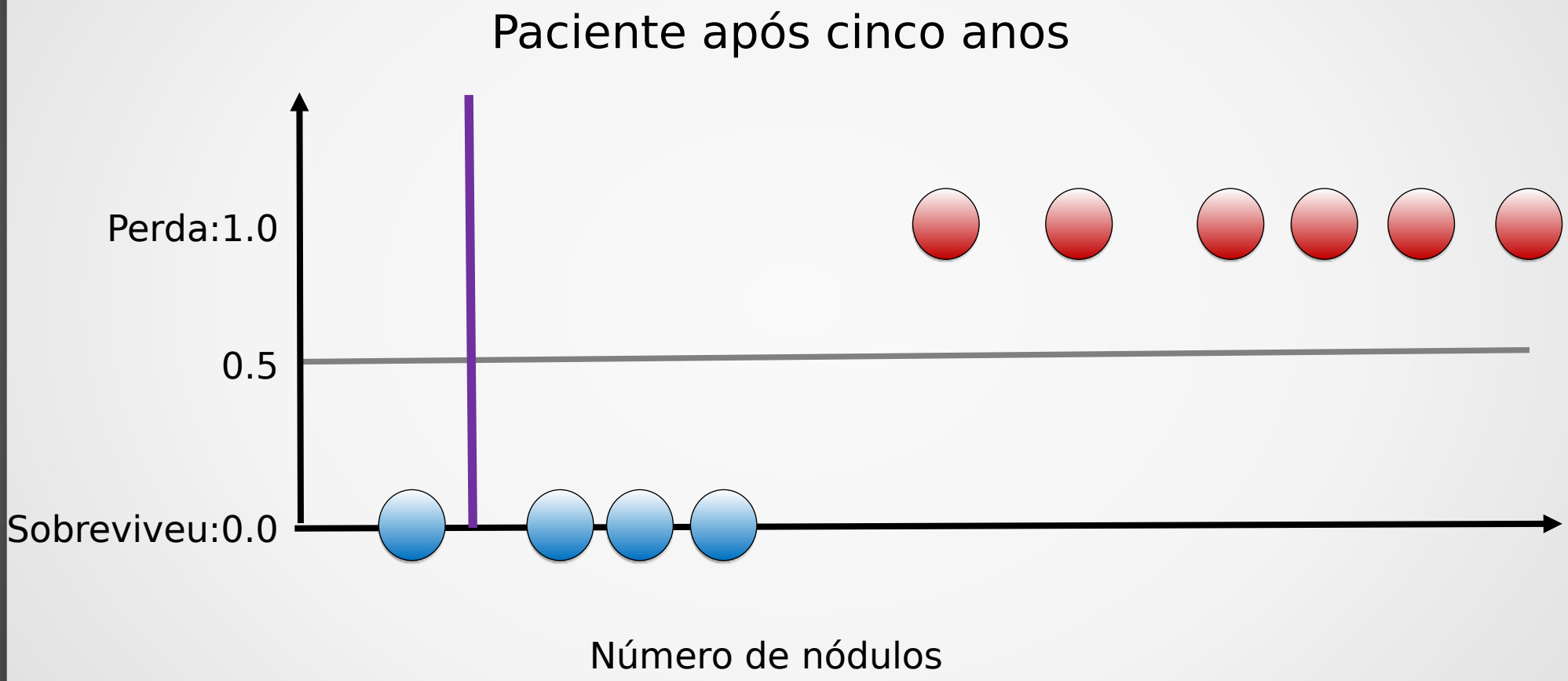
Máquinas de vetores de suporte

- Classificador linear binário não probabilístico
- Objetiva encontrar um separador entre as classes
  - Hiperplano
  - Maximizar a distância entre os pontos mais próximos das classes

Máquinas de Vetores de Suporte (SVM)

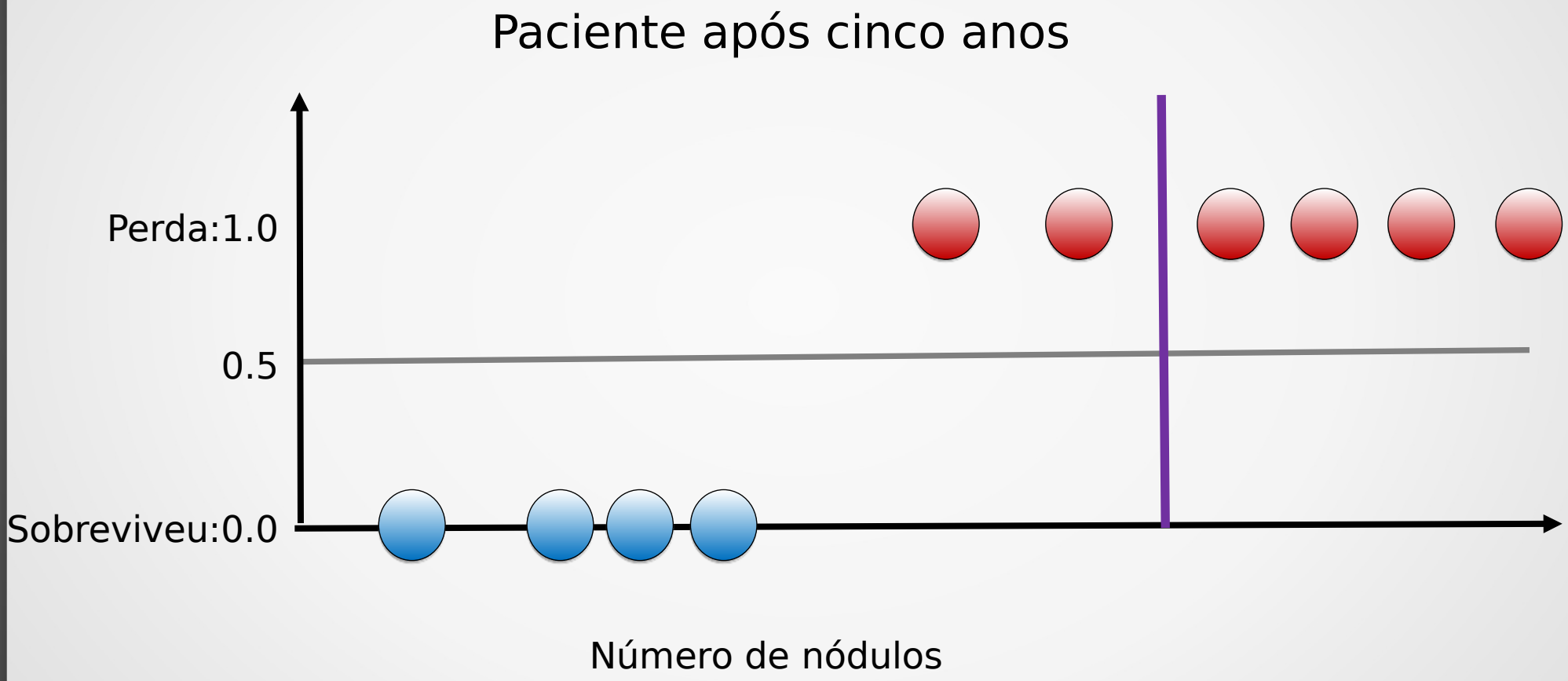


Máquinas de Vetores de Suporte (SVM)



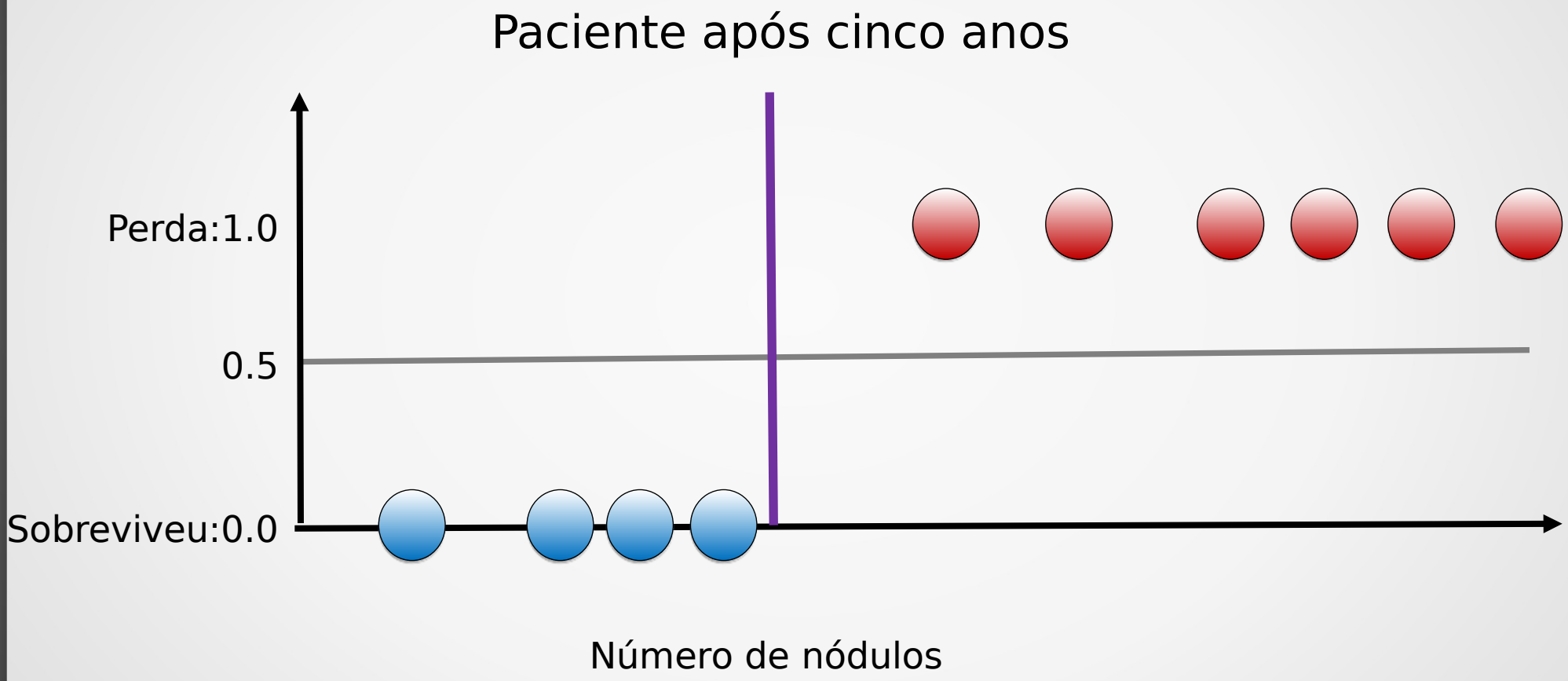
Três erros de classificação

Máquinas de Vetores de Suporte (SVM)



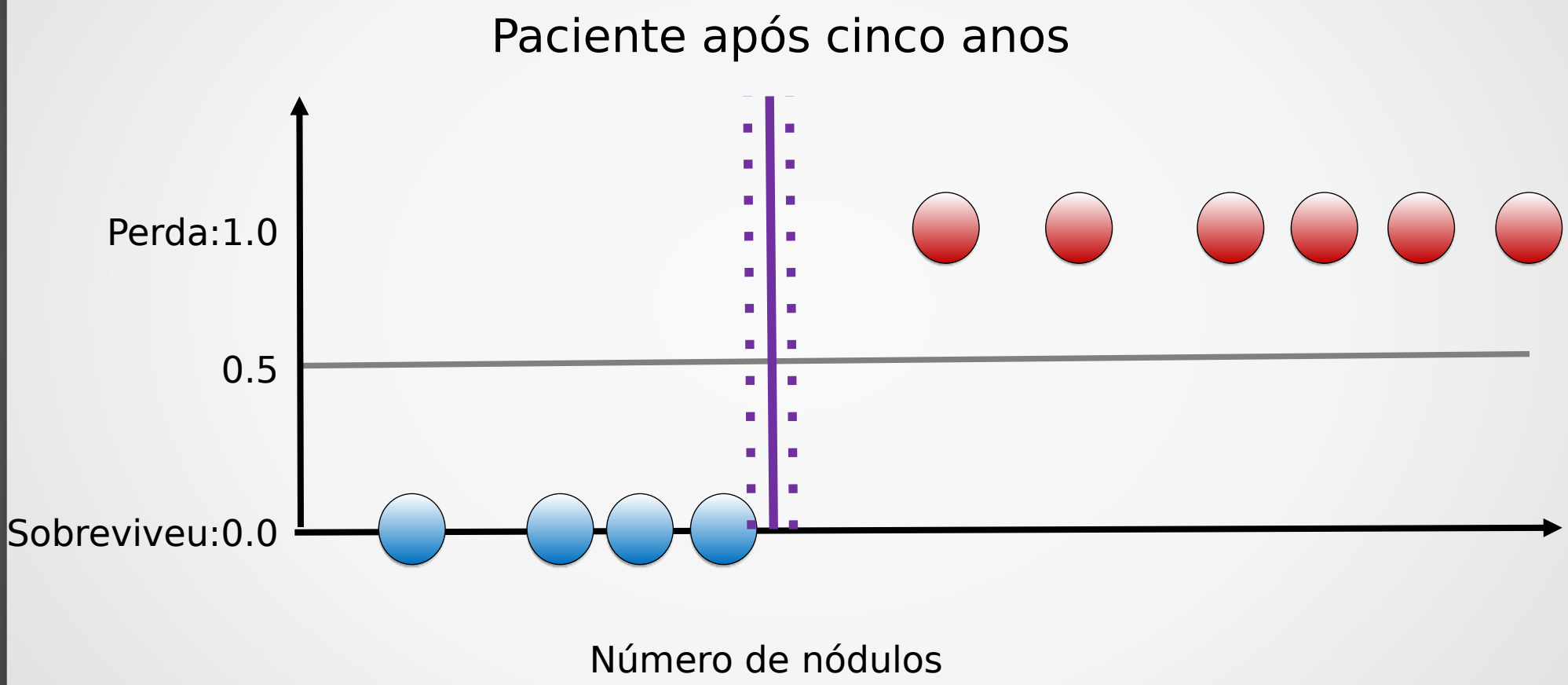
Dois erros de classificação

Máquinas de Vetores de Suporte (SVM)



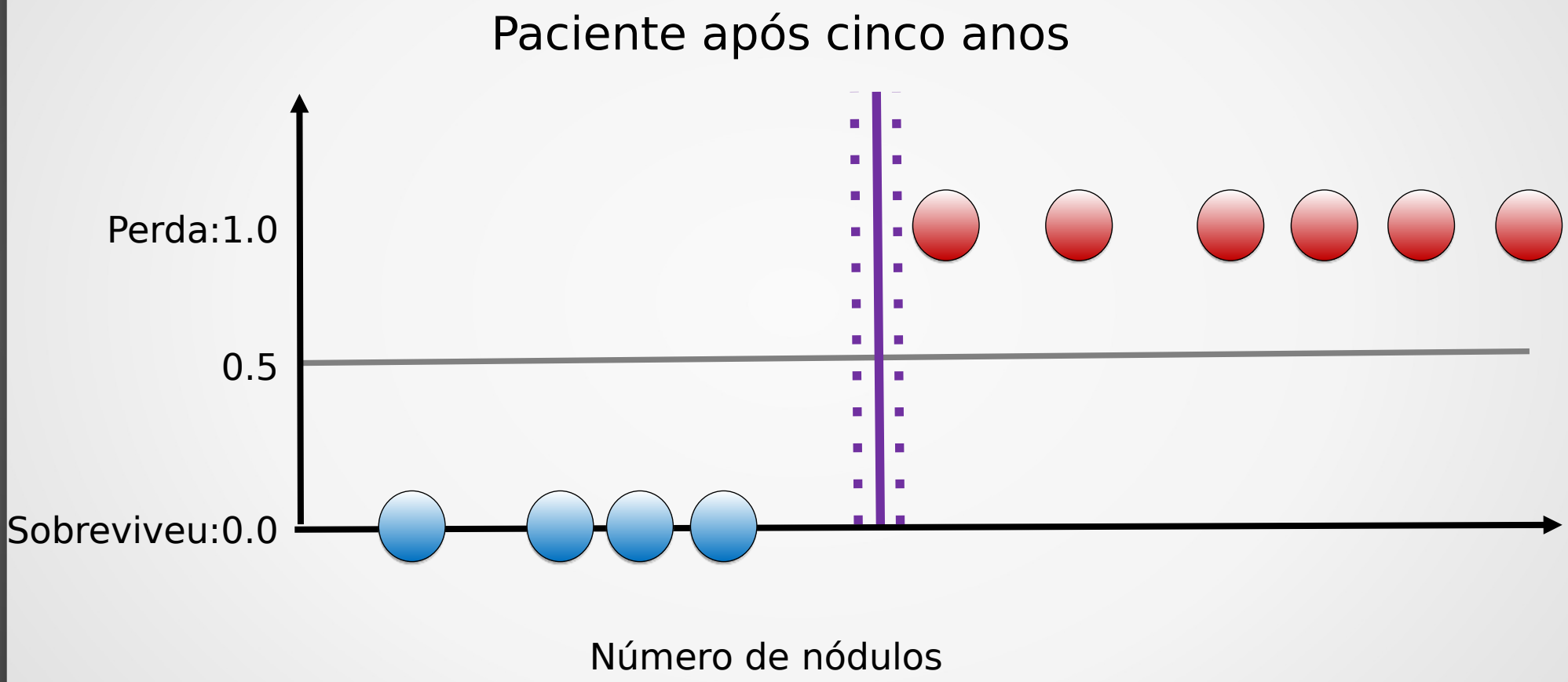
Nenhum erro de classificação

Máquinas de Vetores de Suporte (SVM)



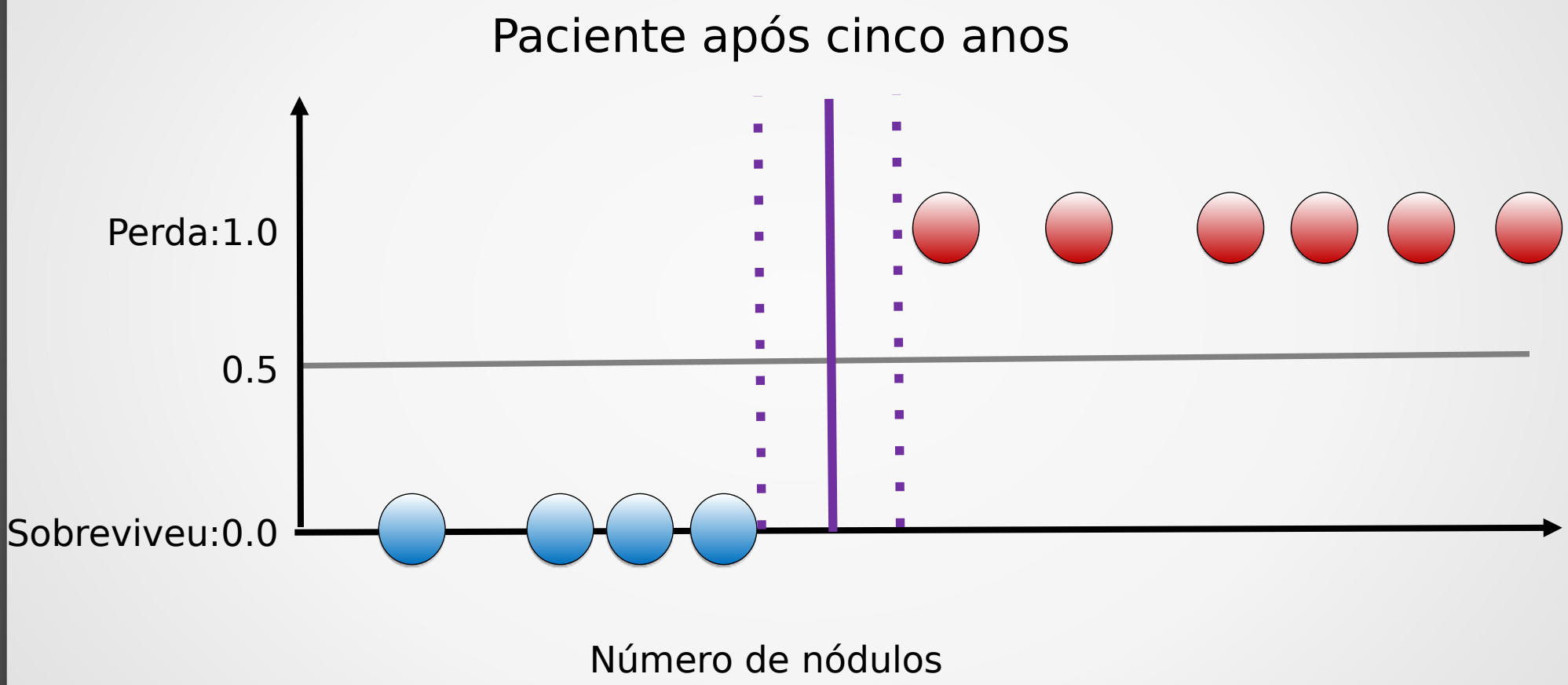
Nenhum erro de classificação: mas é a melhor posição possível?

Máquinas de Vetores de Suporte (SVM)



Nenhum erro de classificação: mas é a melhor posição possível?

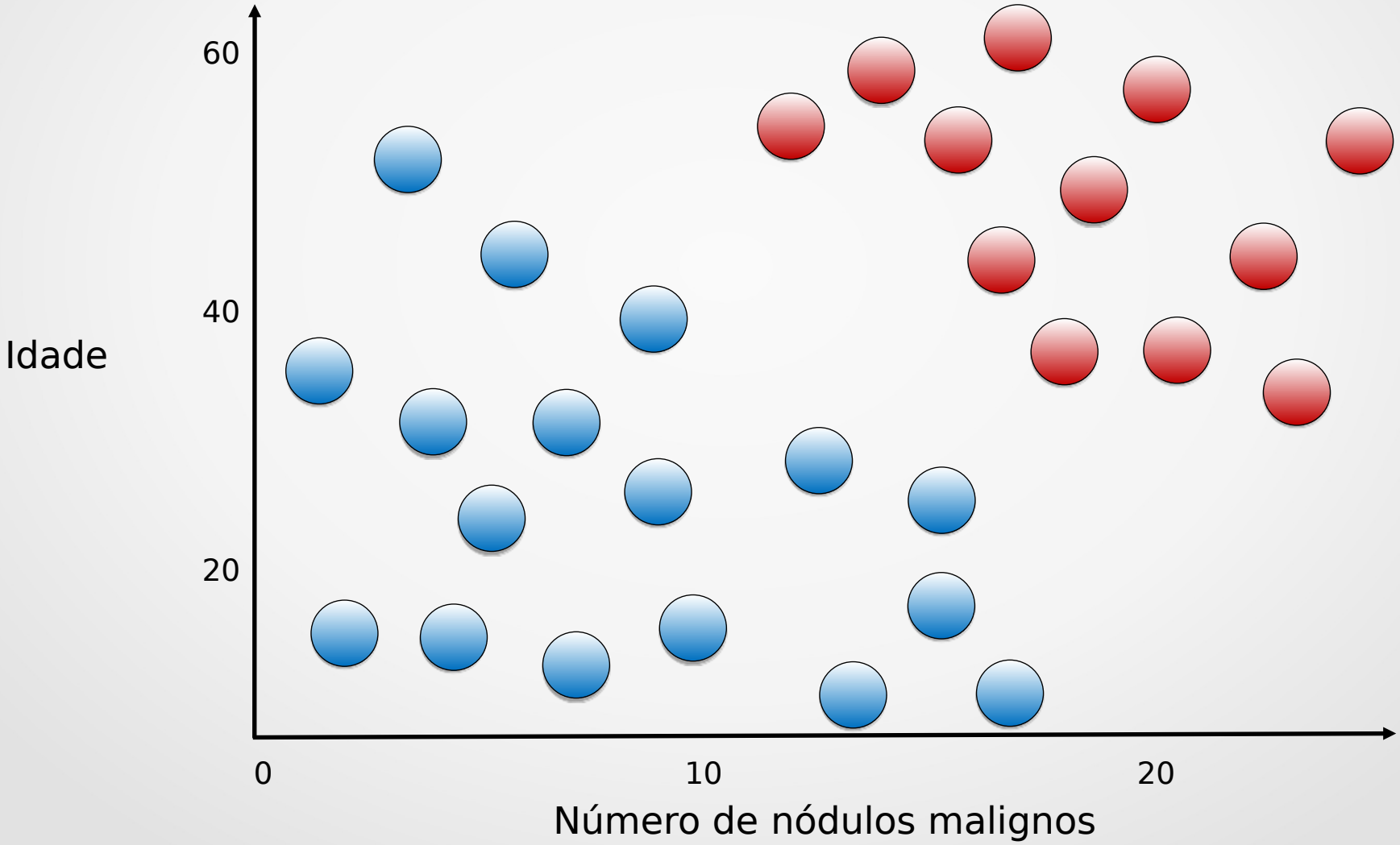
Máquinas de Vetores de Suporte (SVM)



Nenhum erro de classificação: maximizar a região entre classes!

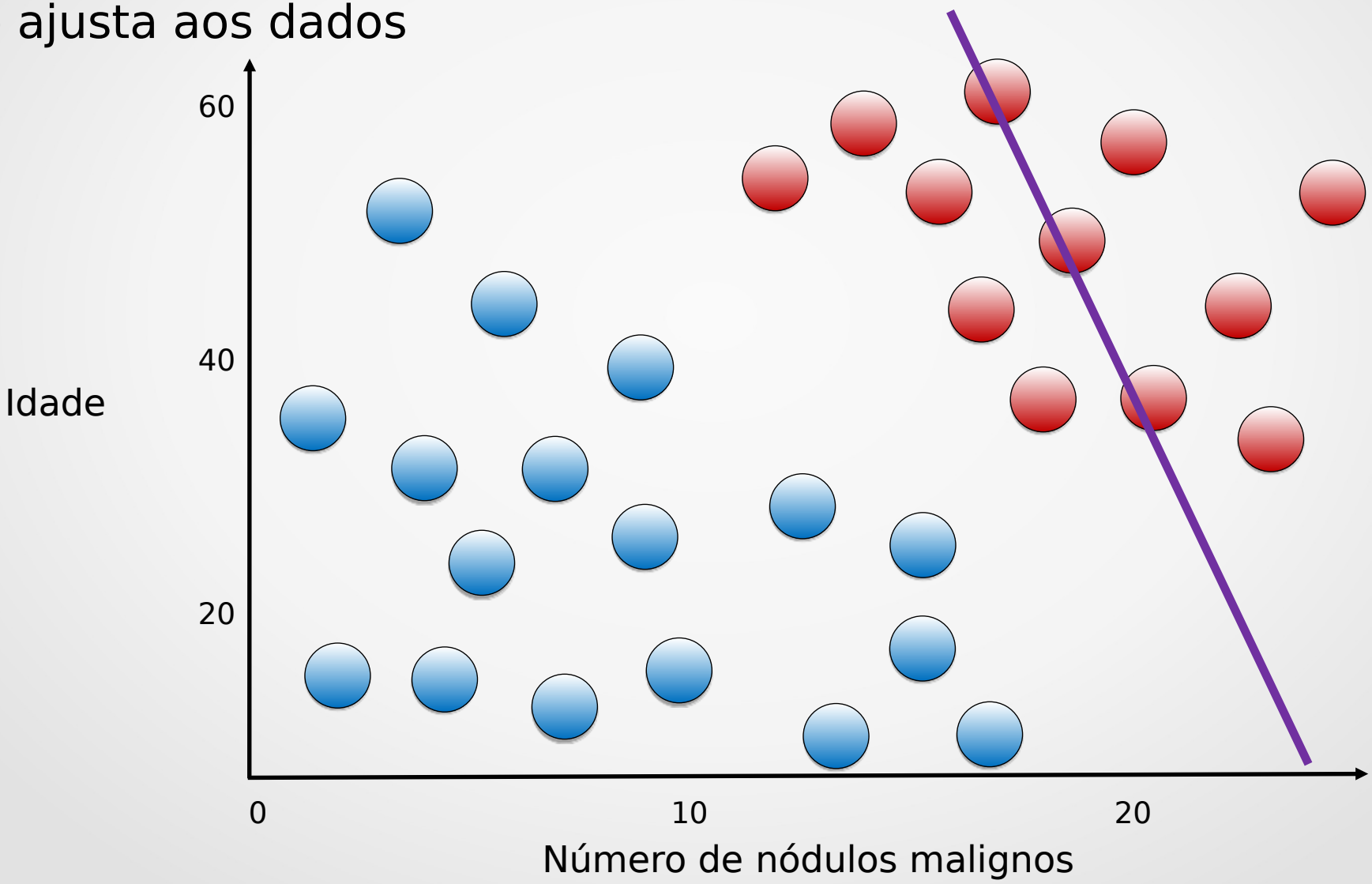
Classificação com SVM

Dois atributos (nódulos, idade)
Duas classes (sobreviveu, perda)



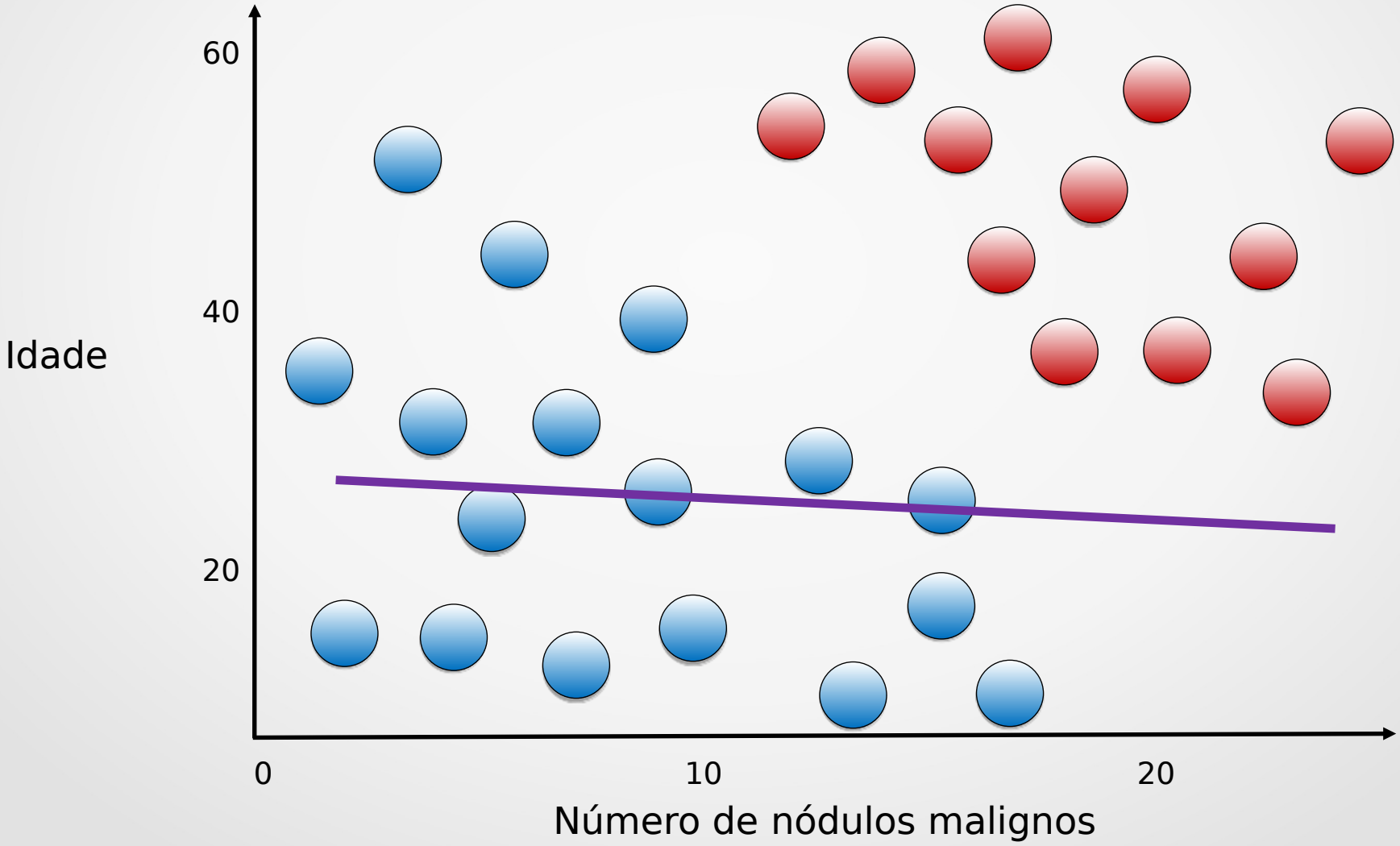
Classificação com SVM

Encontrar a separação linear que melhor se ajusta aos dados



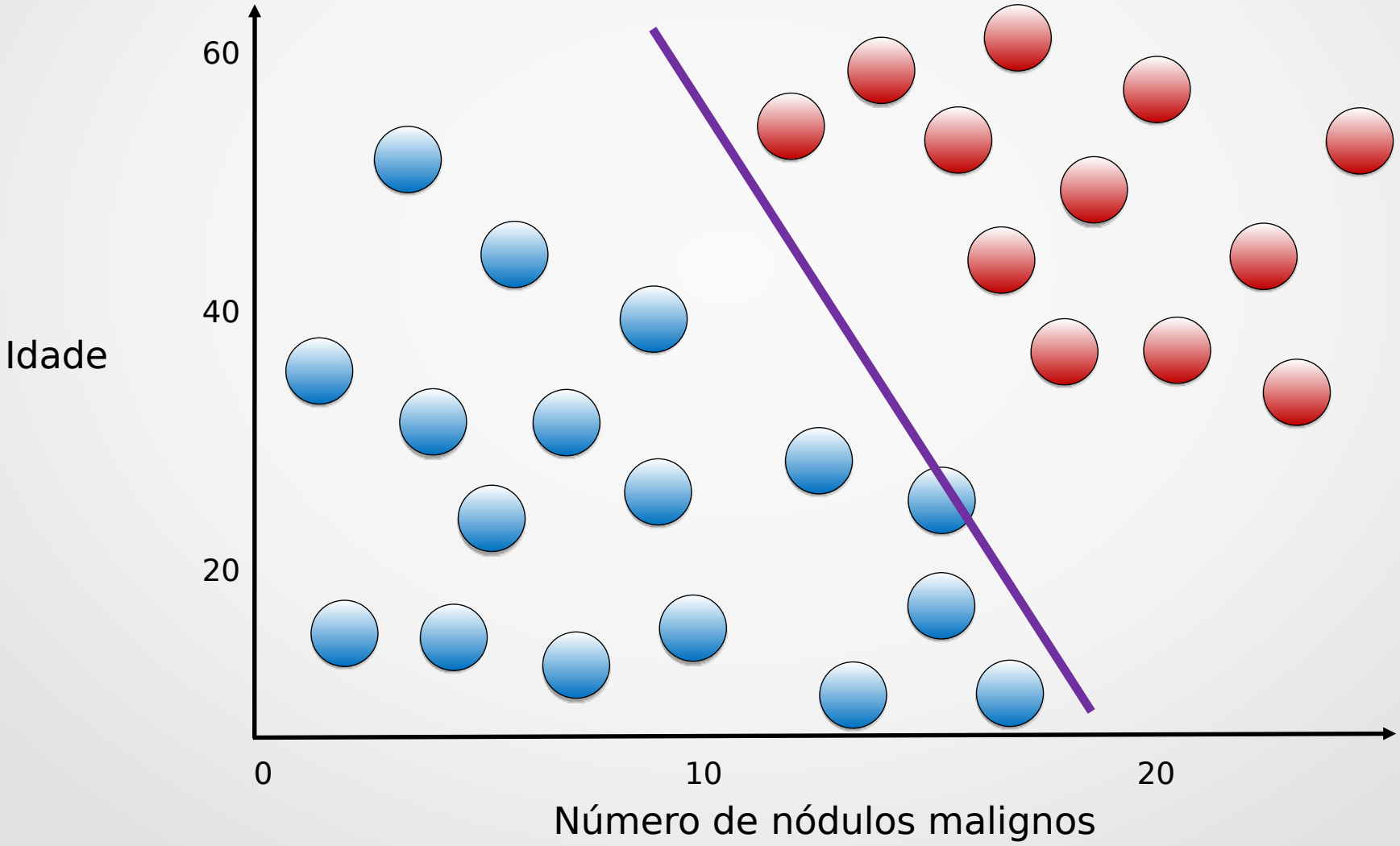
Classificação com SVM

Encontrar a separação linear que melhor se ajusta aos dados



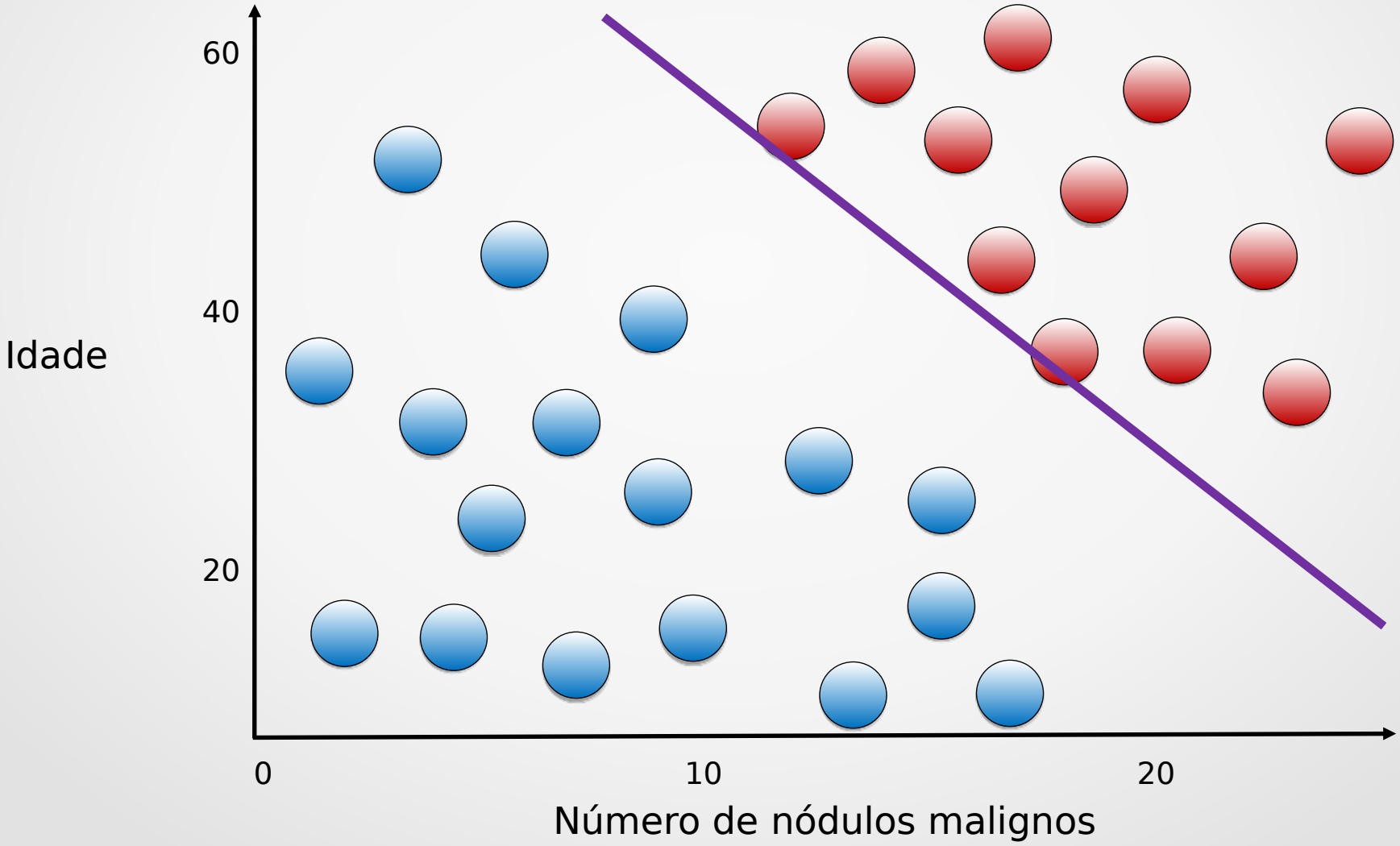
Classificação com SVM

Encontrar a separação linear que melhor se ajusta aos dados



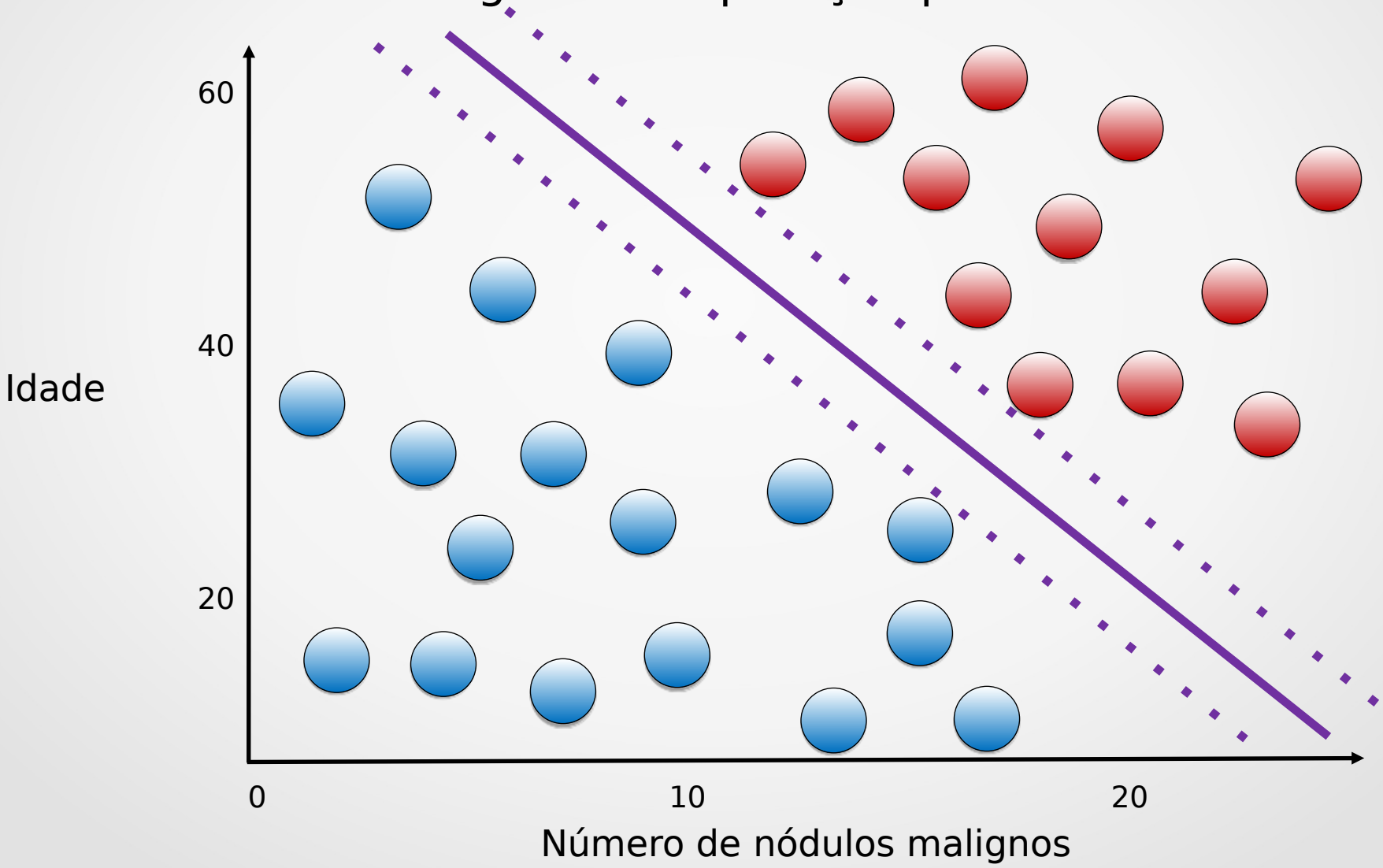
Classificação com SVM

Encontrar a separação linear que melhor se ajusta aos dados



Classificação com SVM

Que inclui a maior margem de separação possível!



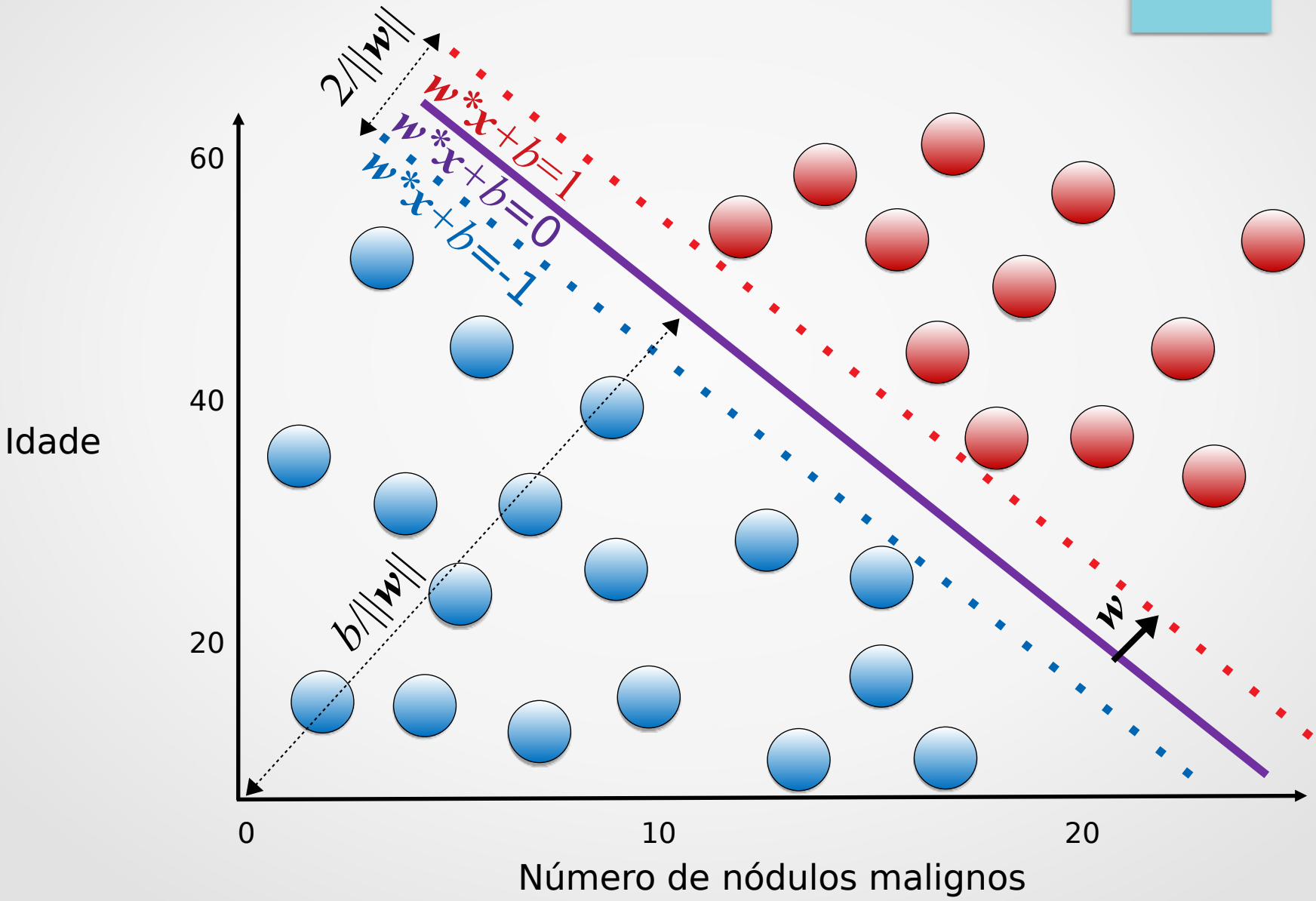
Matemática por trás do SVM

- Temos as seguintes considerações:
  - Y ∈ {-1,+1}
  - X é linearmente separável (por um hiperplano)

$$h(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b$$

  - em que w é uma vetor normal ao hiperplano, não necessariamente unitário, b/|**w**| é a distância do hiperplano em relação a origem no sentido do vetor normal **w**.

Matemática por trás do SVM

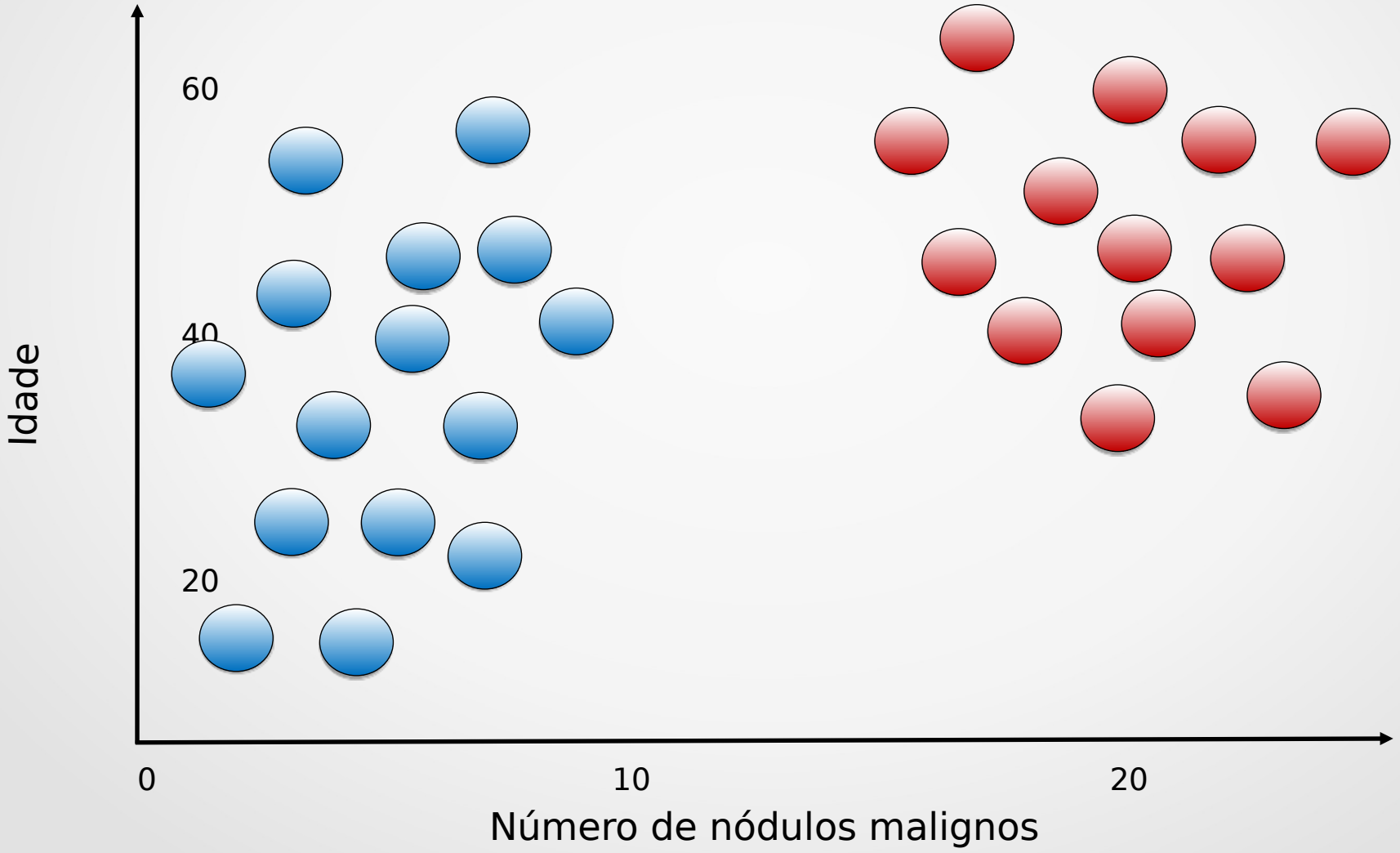


SVM (linear) com margens rígidas

- Geometricamente, a distância entre a duas margens (hiperplanos) é 2/||$\boldsymbol{w}$||
  - Portanto, para maximizar a distância entre as margens, o problema resume-se em minimizar ||$\boldsymbol{w}$||
    - Restrito a $y_i(w*x_i+b)$>=1, para $i = 1,\ldots,n$
      - O que garante que não há exemplos entre as margens!
  - A solução é encontrar $\boldsymbol{w}$ e b que solucionam esse problema

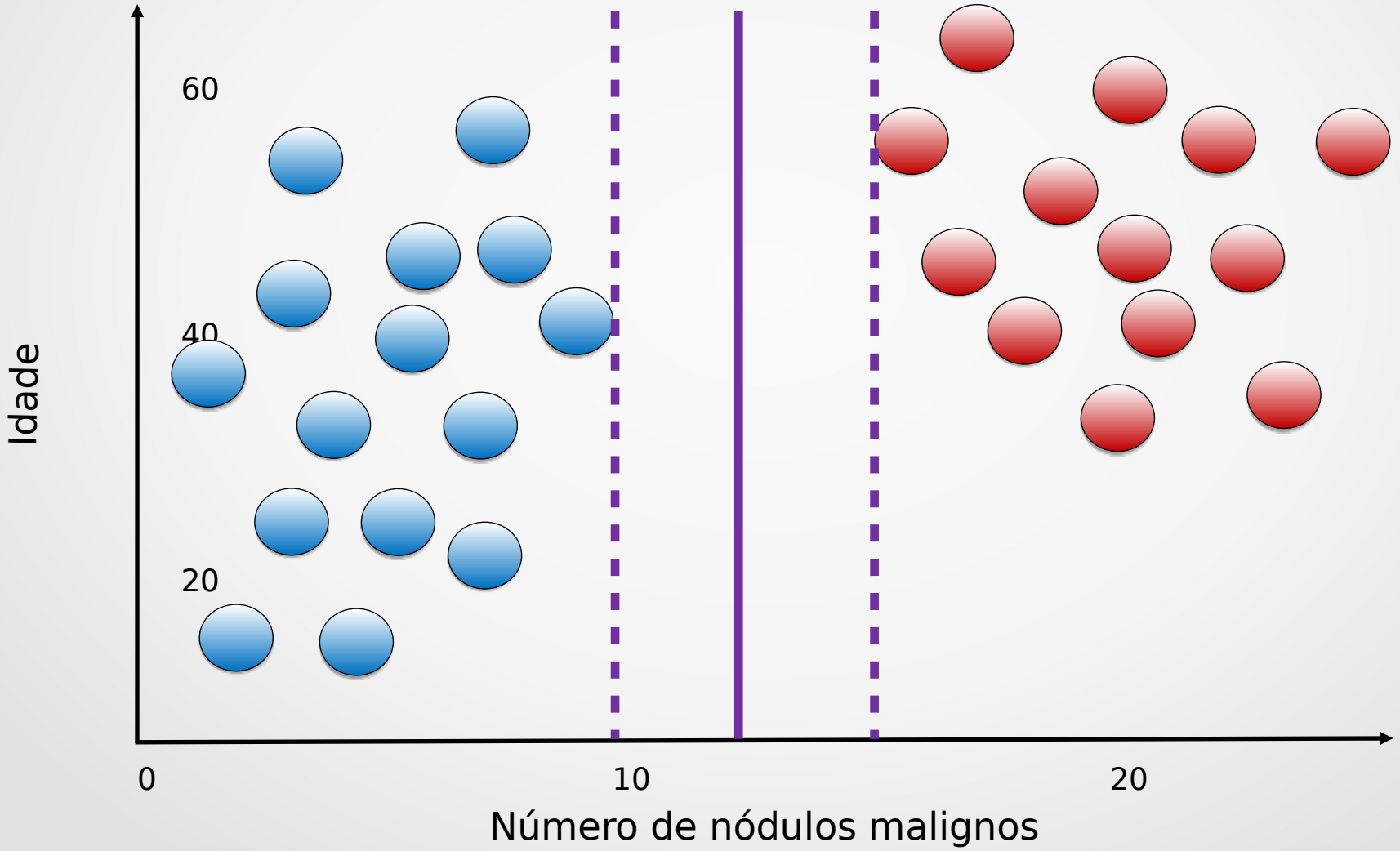
Sensibilidade a externos (*outliers*)

- SVMs apresentam sensibilidade a *outliers*



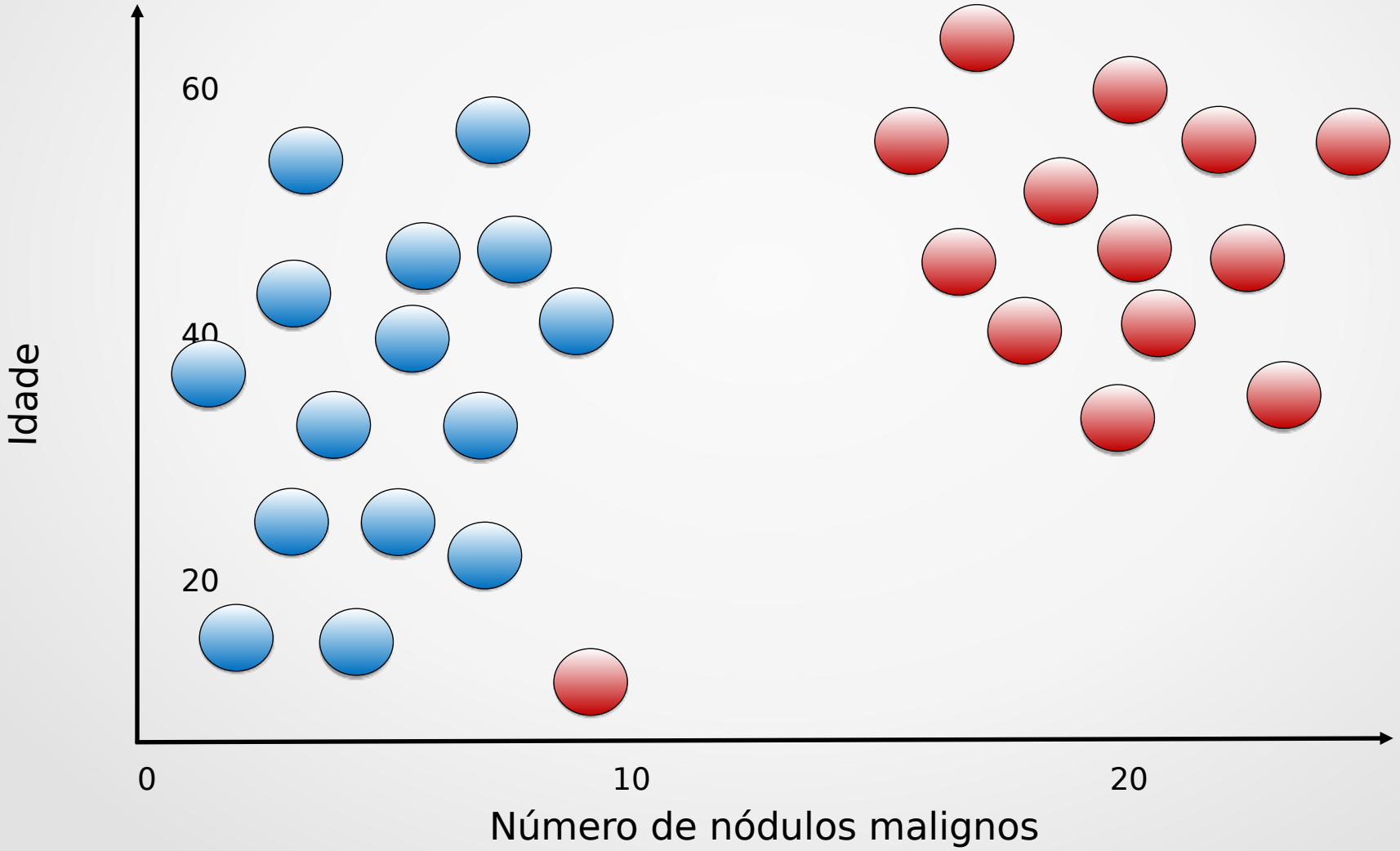
Sensibilidade a externos (*outliers*)

- SVMs apresentam sensibilidade a *outliers*



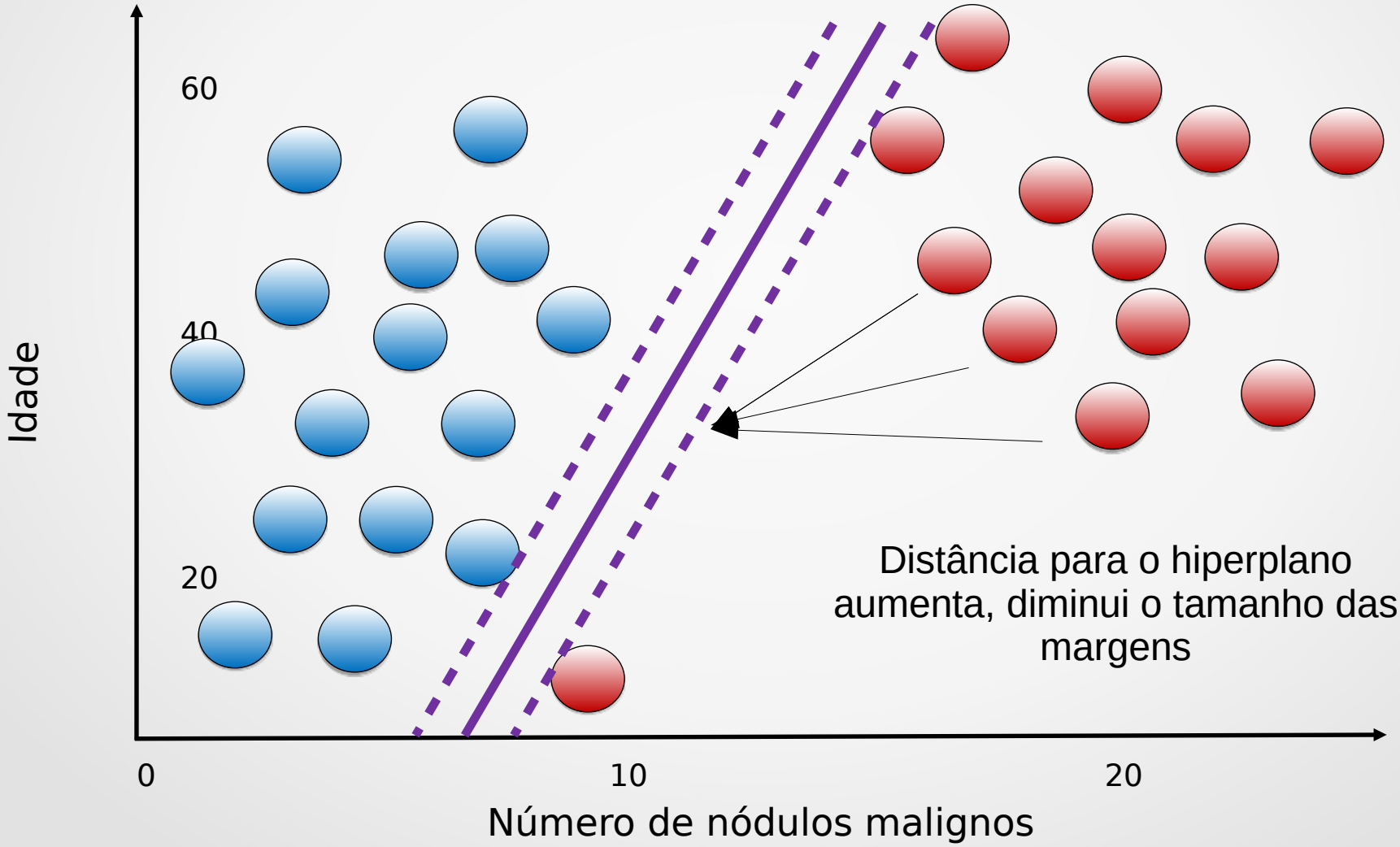
Sensibilidade a externos (*outliers*)

- SVMs apresentam sensibilidade a *outliers*



Sensibilidade a externos (*outliers*)

- SVMs apresentam sensibilidade a *outliers*



SVM (linear) com margens suaves

- Para lidar melhor com ruídos, *borderlines* e *outliers*, etc, é preciso suavizar as restrições de margens do SVM

$$y_i(\mathbf{w} \cdot \mathbf{x} + b) \geq 1 - \xi_i, \xi_i \geq 0, \forall i = 1 \ldots n)$$

$$\text{Minimizar: } \frac{1}{2}||\mathbf{w}||^2 + C(\sum_{i=1}^{n} \xi_i)$$

- “A constante C é um termo de regularização que impõe um peso à minimização dos erros no conjunto de treinamento em relação à minimização da complexidade do modelo” - Faceli et al., 2011
  - Quanto maior, maior o peso dos erros. Ou seja, C grande significa margem pequena.

Aplicando SVM linear com holdout Iris

```
import pandas as pd
#Importa SVM linear do Sklearn
from sklearn.svm import LinearSVC
# Localização do arquivo
filepath = 'data/Iris_Data.csv'
# Importando os dados
data = pd.read_csv(filepath)
#Colocando os dados em ordem aleatória
randomdata = (data.sample(n=150, replace=False))
#Aplicando hold out
traindata = randomdata.iloc[:135,:]
testdata = randomdata.iloc[135:,:]
#Cria uma instância de classe
LinSVC = LinearSVC(penalty='l2', C=10.0)
#Ajusta o modelo SVM aos dados de treino
LinSVC = LinSVC.fit(traindata.iloc[:,0:4], traindata.iloc[:,4])
#Classe real
print(testdata.iloc[:,4])
#Classe predita
print(LinSVC.predict(testdata.iloc[:,0:4]))
```

Aplicando SVM linear com holdout Iris

- Saída = Classes

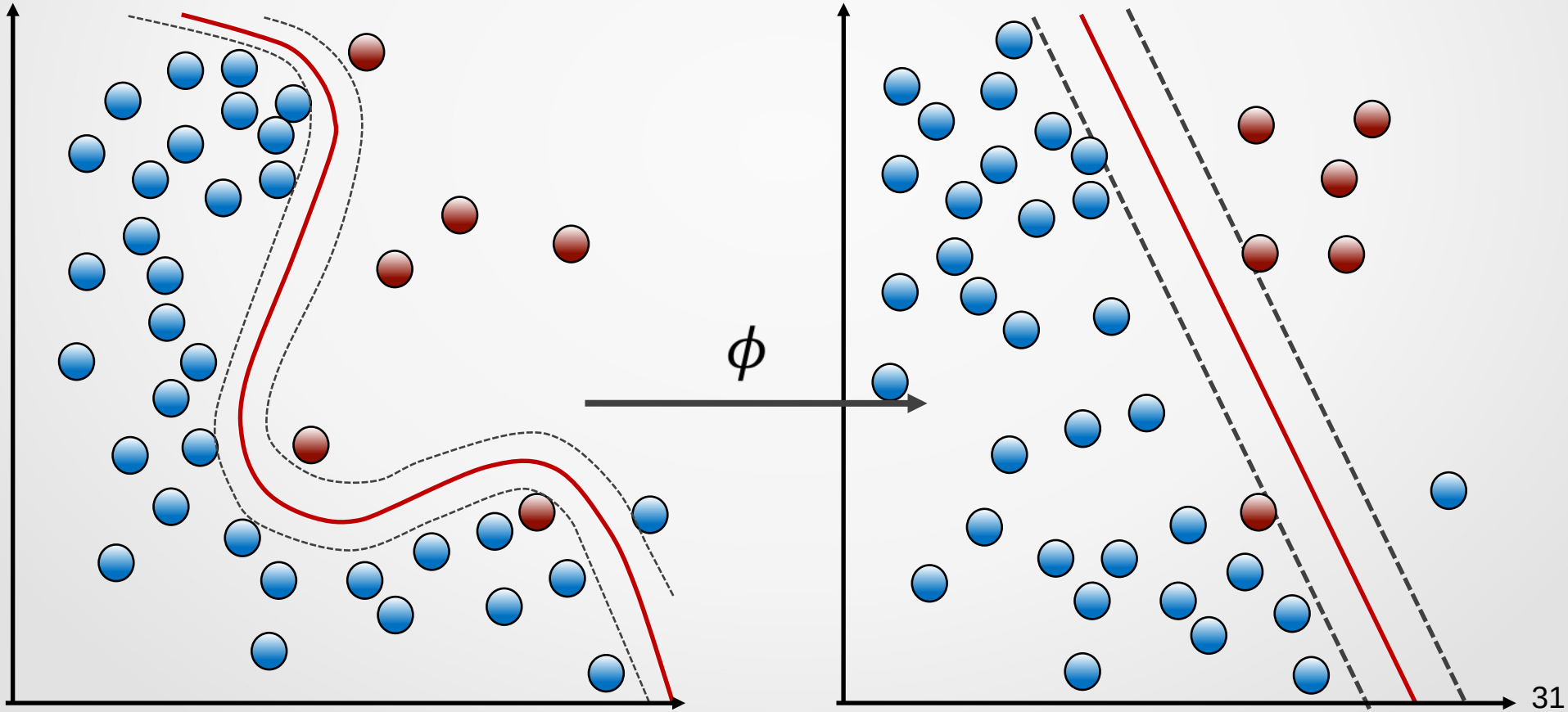
```
132      Iris-virginica
50     Iris-versicolor
6          Iris-setosa
113      Iris-virginica
13         Iris-setosa
30         Iris-setosa
18         Iris-setosa
61     Iris-versicolor
104      Iris-virginica
62     Iris-versicolor
87     Iris-versicolor
123      Iris-virginica
149      Iris-virginica
28         Iris-setosa
140      Iris-virginica
```

- Saída Predita

```
['Iris-virginica'
'Iris-versicolor'
'Iris-setosa'
'Iris-virginica'
 'Iris-setosa'
'Iris-setosa'
'Iris-setosa'
'Iris-versicolor'
 'Iris-virginica'
'Iris-versicolor'
'Iris-versicolor'
'Iris-versicolor'
 'Iris-virginica'
'Iris-setosa'
'Iris-virginica']
```

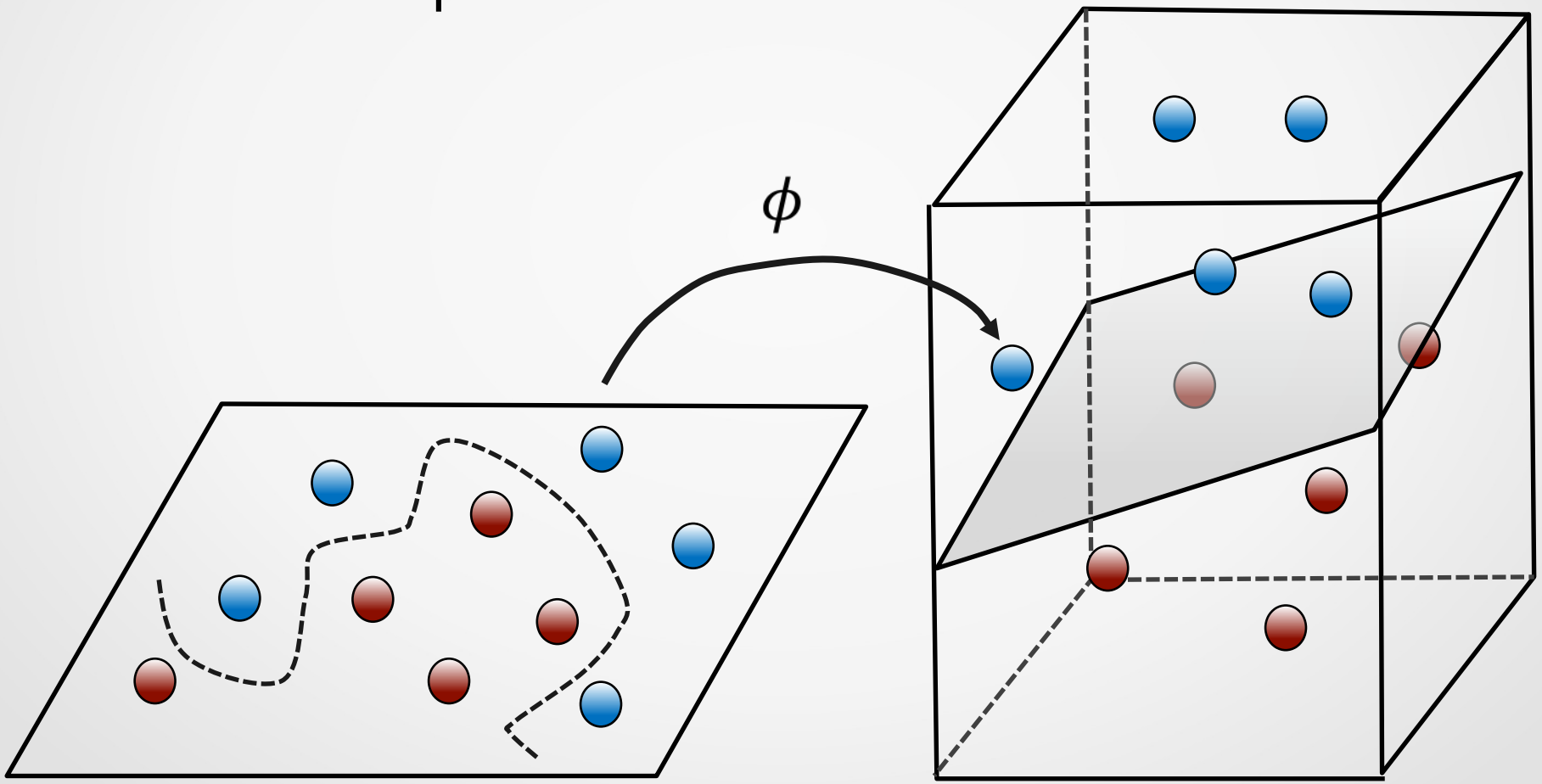
Margens de decisão não linear

- Dados que possuem divisão não linear entre classes podem ser separadas linearmente em espaços com maior dimensionalidade



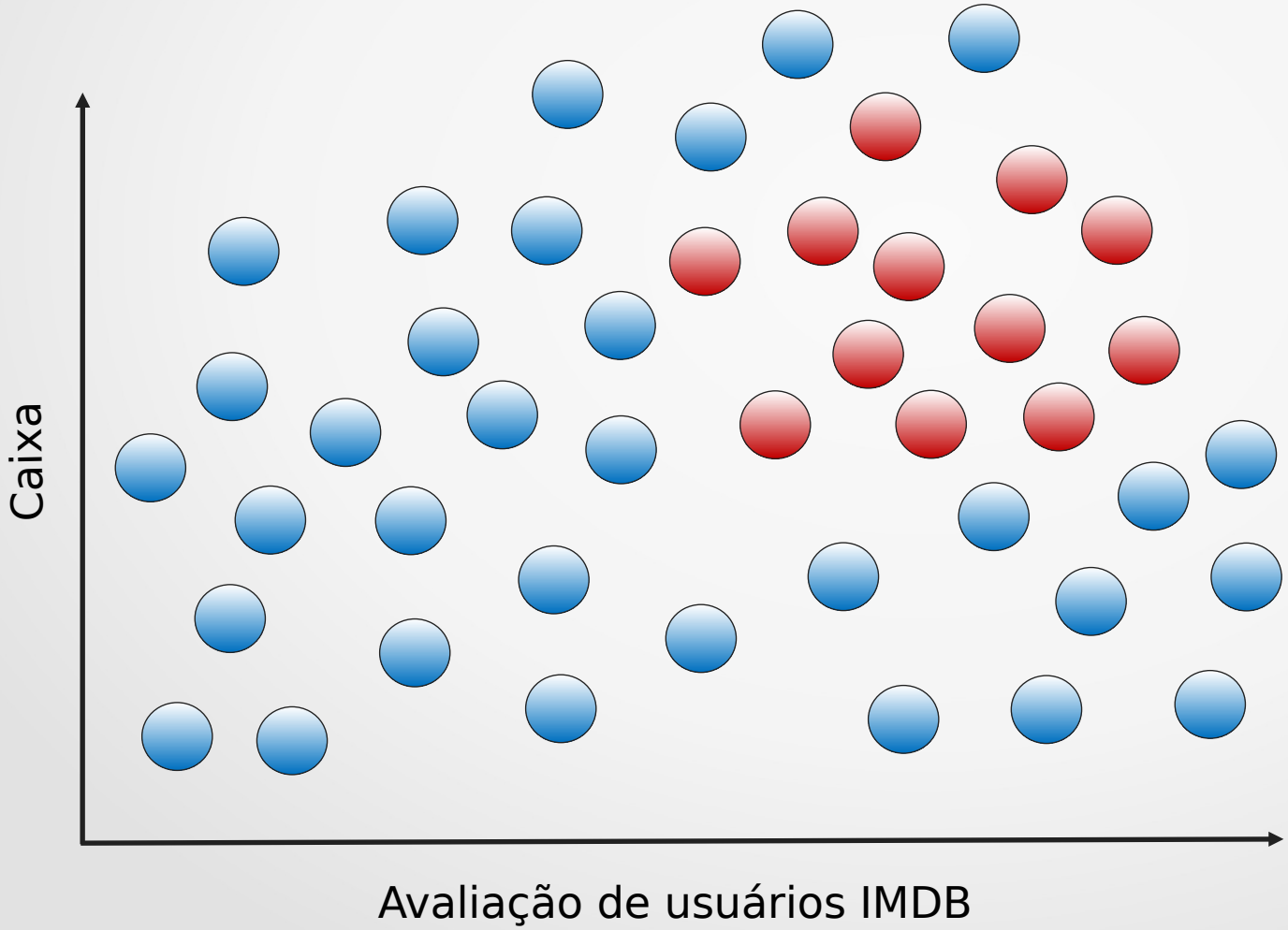
Truque de Kernel

- Transformar o espaço de forma que os dados tornem-se linearmente separáveis



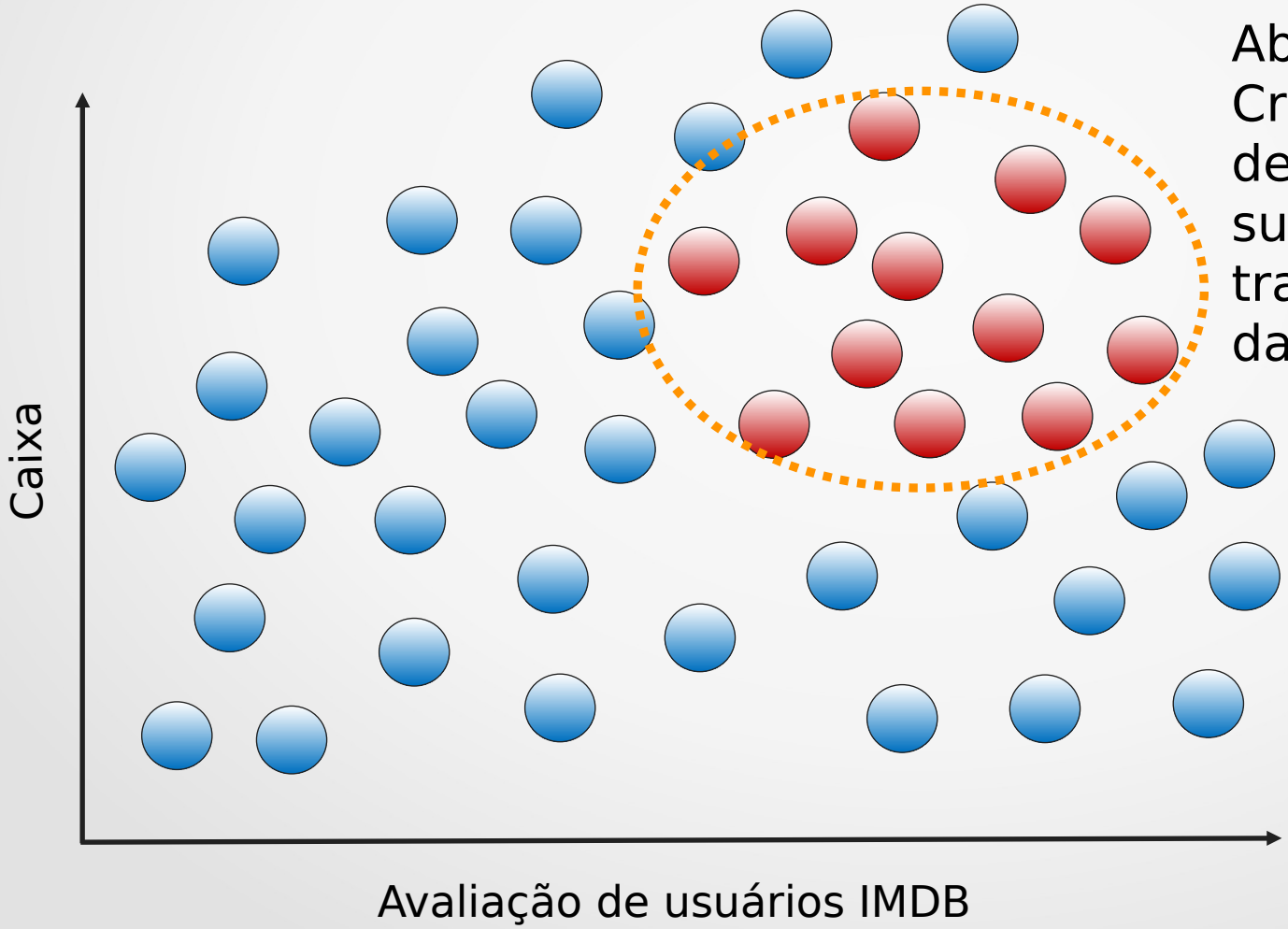
Exemplo uso de Kernel Gaussiano

- Vencedores do Palme d'Or do festival de Cannes



Exemplo uso de Kernel Gaussiano

- Vencedores do Palme d'Or do festival de Cannes

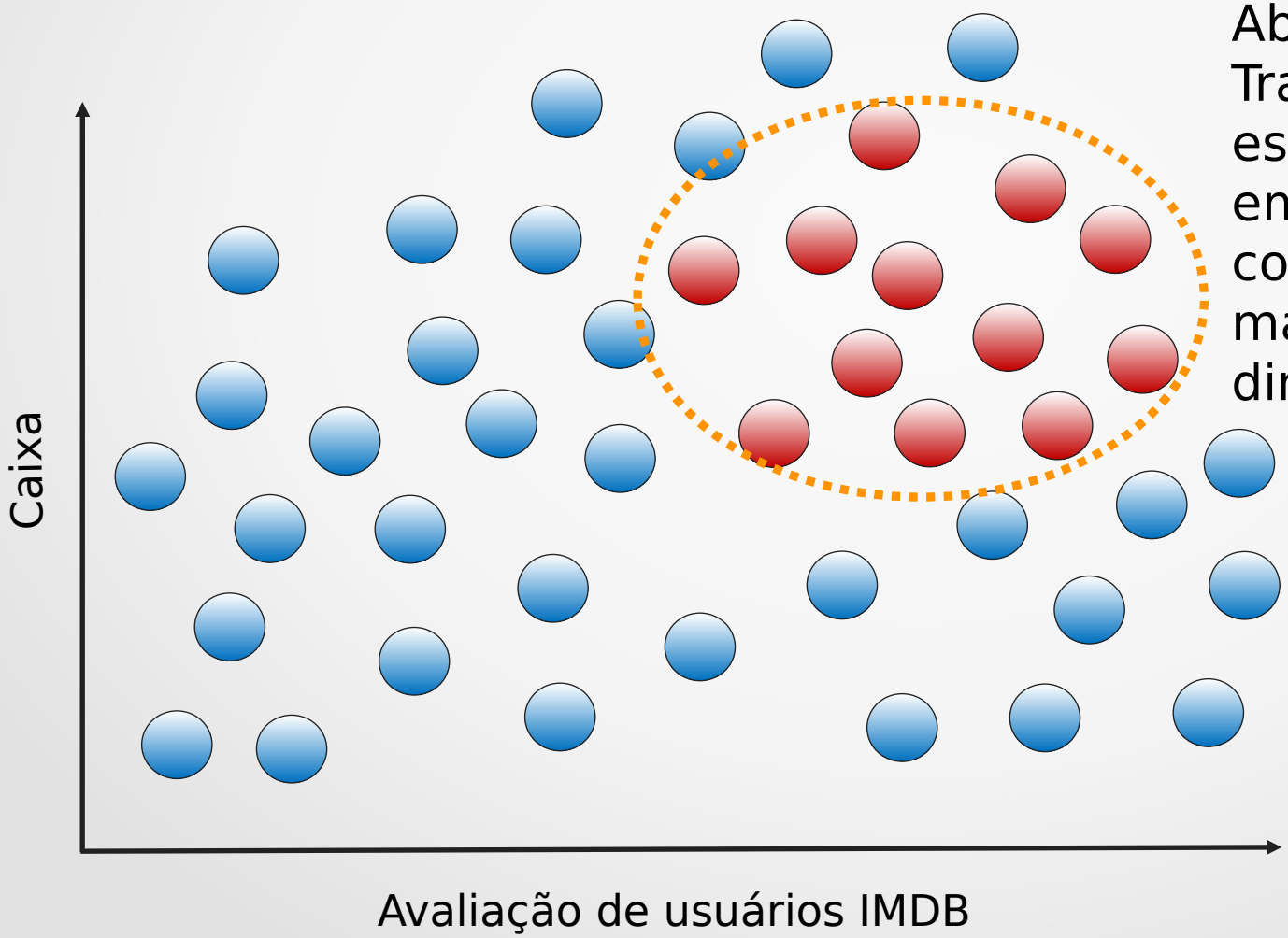


Abordagem 1: Criar atributos de ordem superior para transformar os dados.

$Caixa^2$ +
$Avaliação^2$ +
Caixa *
Avaliação +
...

Exemplo uso de Kernel Gaussiano

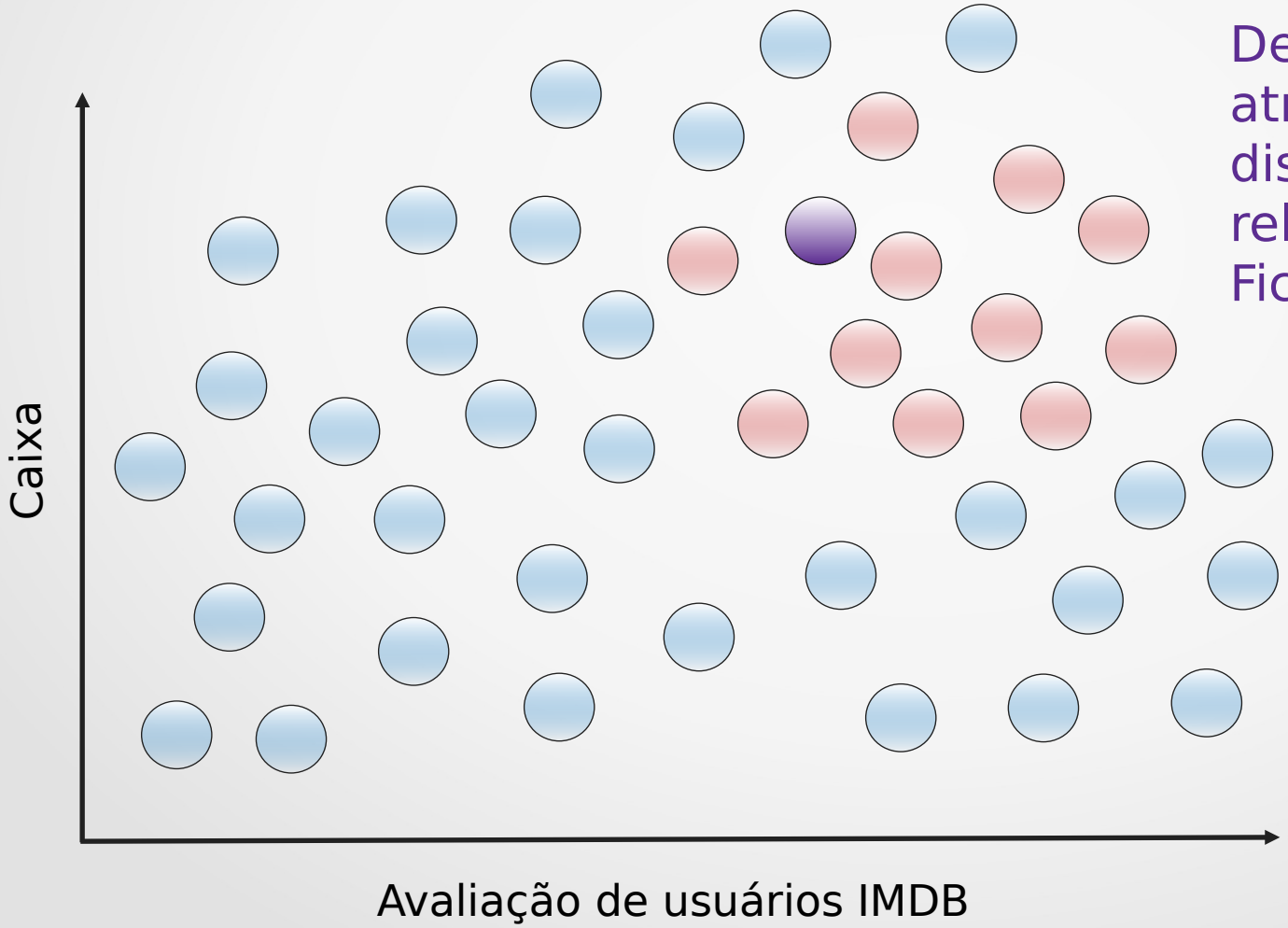
- Vencedores do Palme d'Or do festival de Cannes



Abordagem 2: Transformar o espaço dos dados em um sistema de coordenadas de maior dimensionalidade

Exemplo uso de Kernel Gaussiano

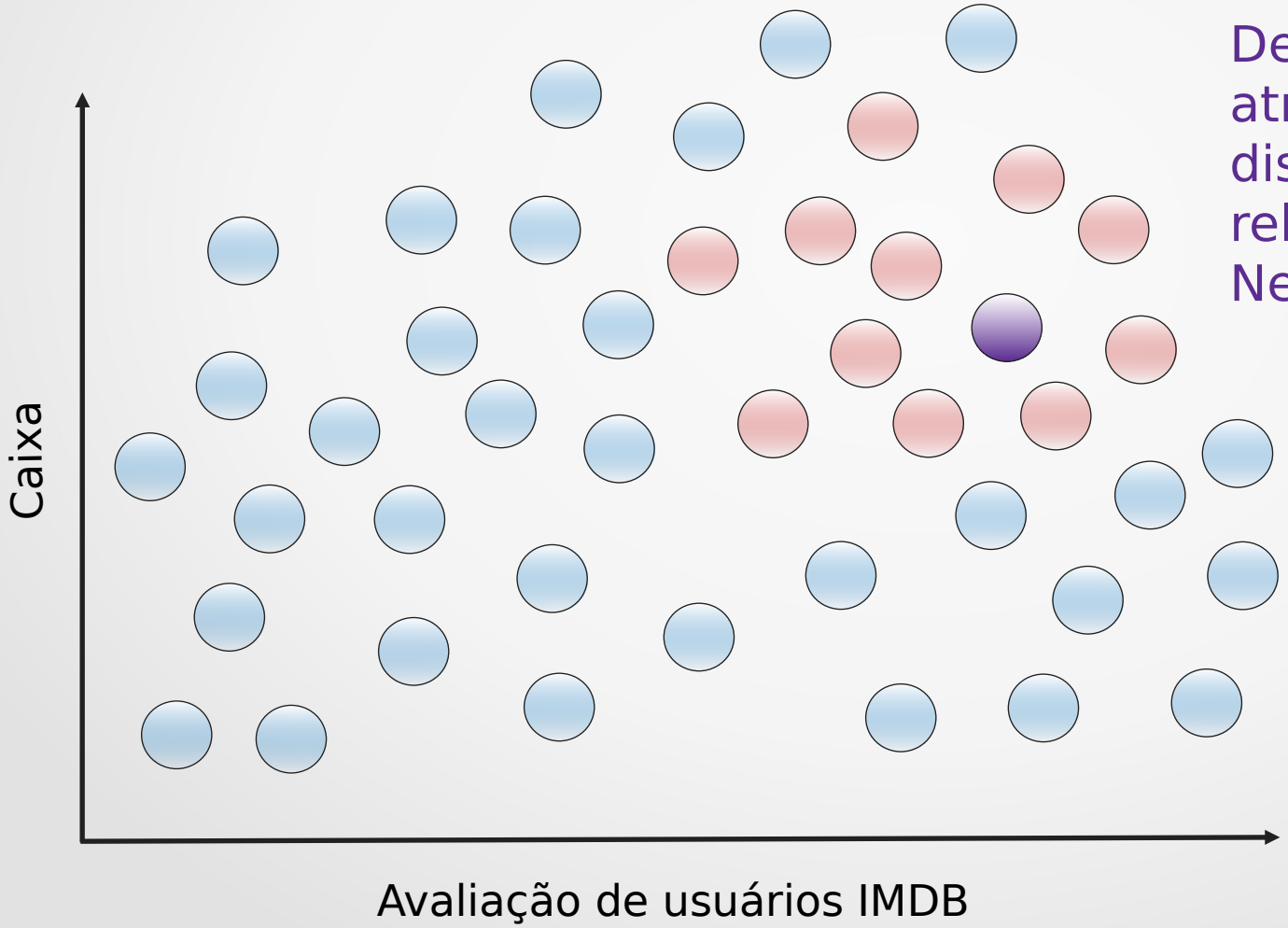
- Vencedores do Palme d'Or do festival de Cannes



Definir novo atributo 1: dissimilaridade em relação a "Pulp Fiction"

Exemplo uso de Kernel Gaussiano

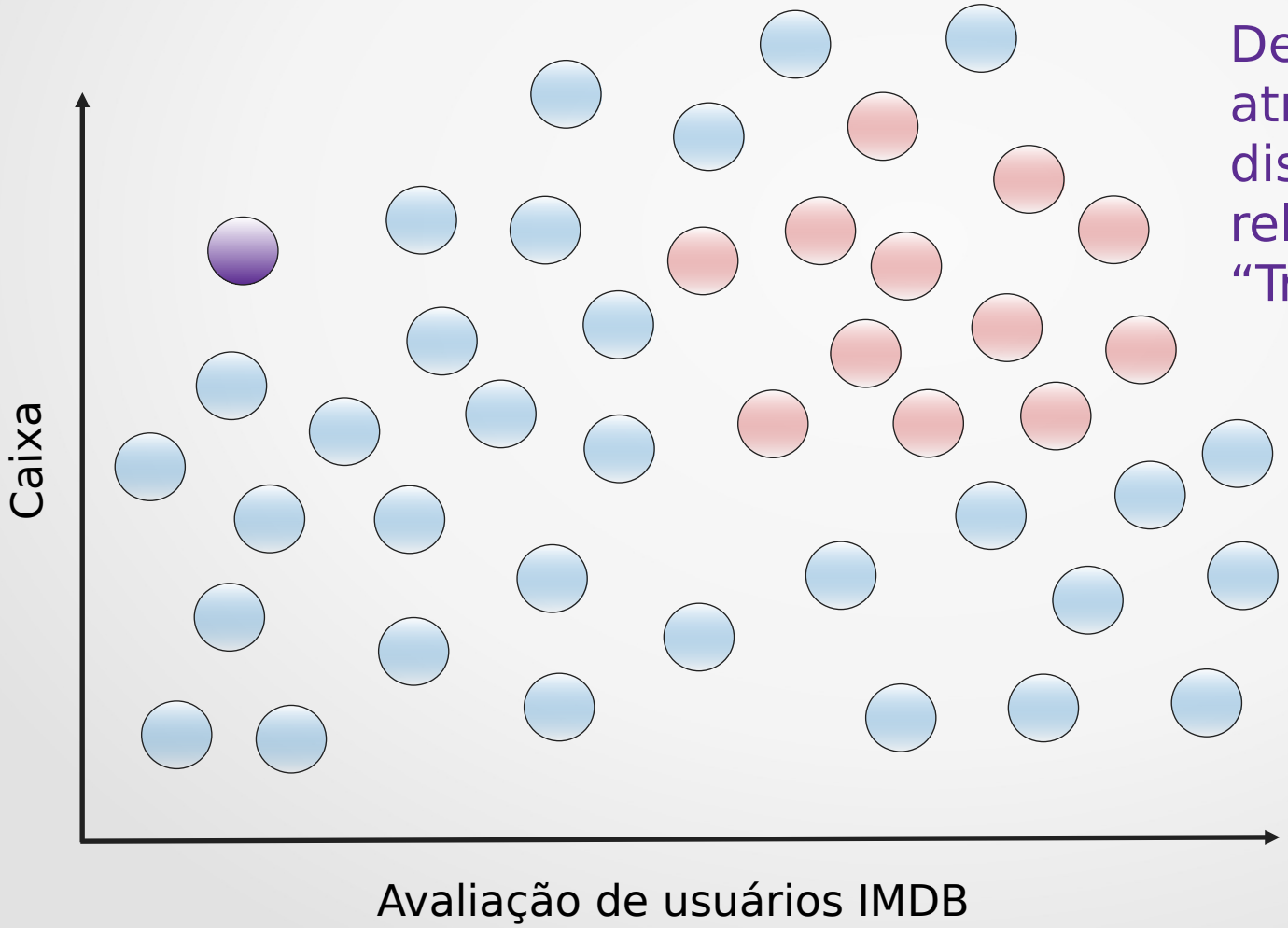
- Vencedores do Palme d'Or do festival de Cannes



Definir novo atributo 2: dissimilaridade em relação a "Cisne Negro"

Exemplo uso de Kernel Gaussiano

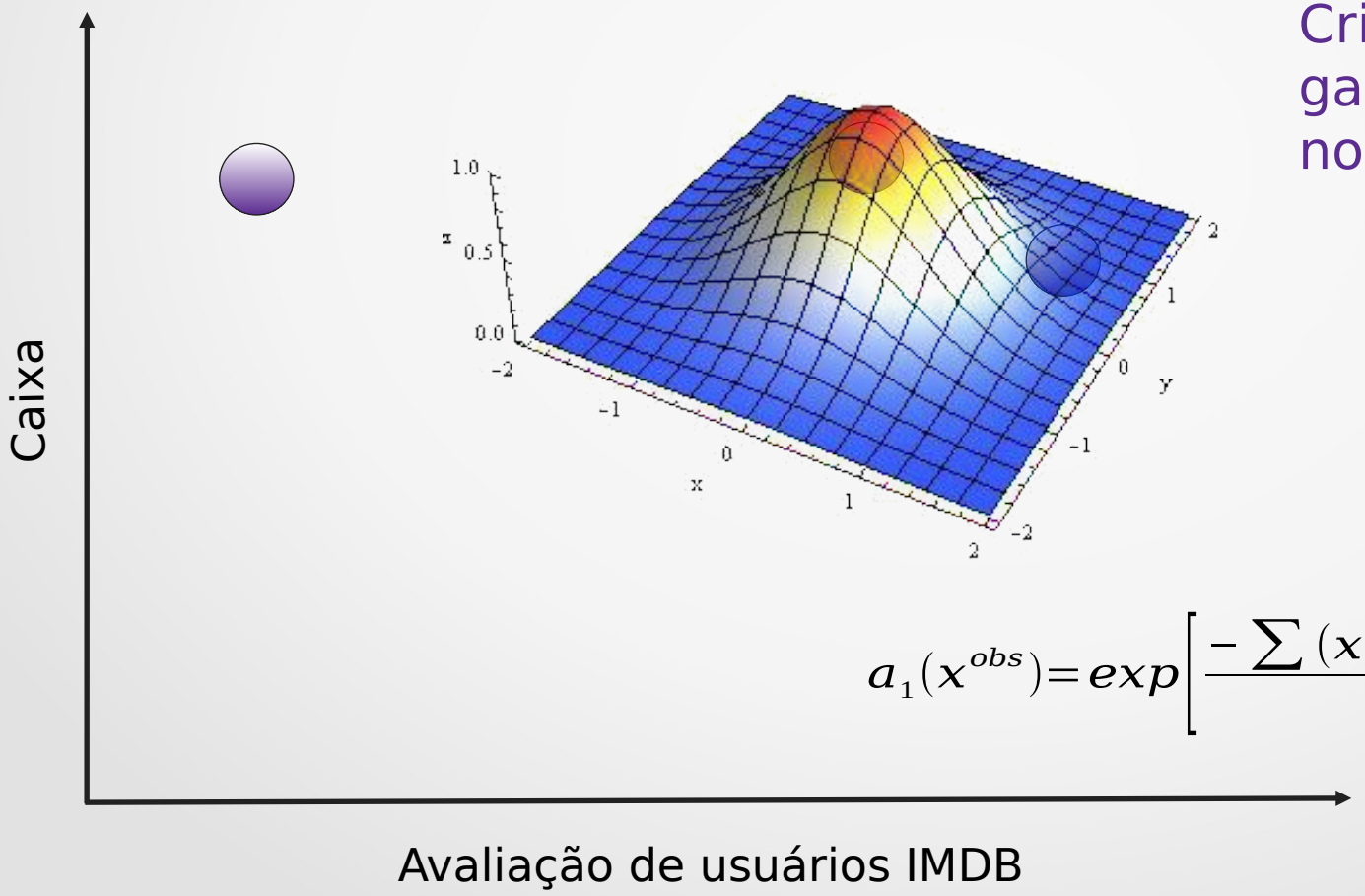
- Vencedores do Palme d'Or do festival de Cannes



Definir novo atributo 3: dissimilaridade em relação a "Transformers"

Exemplo uso de Kernel Gaussiano

- Vencedores do Palme d'Or do festival de Cannes



Criar um função gaussiana sobre o novo atributo 1

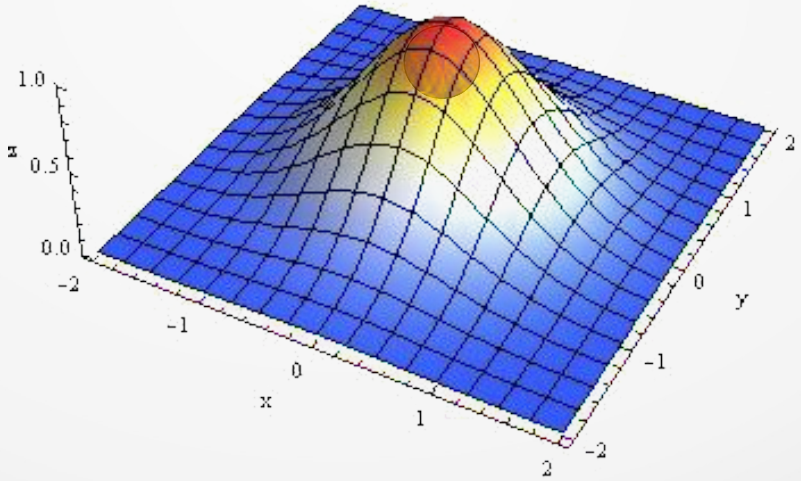
$$a_1(x^{obs}) = exp\left[\frac{-\sum (x_i^{obs} - x_i^{Pulp\ Fiction})^2}{2\sigma^2}\right]$$

Exemplo uso de Kernel Gaussiano

- Vencedores do Palme d'Or do festival de Cannes

$$a_1(x^{obs}) = exp\left[\frac{-\sum (x_i^{obs} - x_i^{BlackSwan})^2}{2\sigma^2}\right]$$

Criar um função gaussiana sobre o novo atributo 2

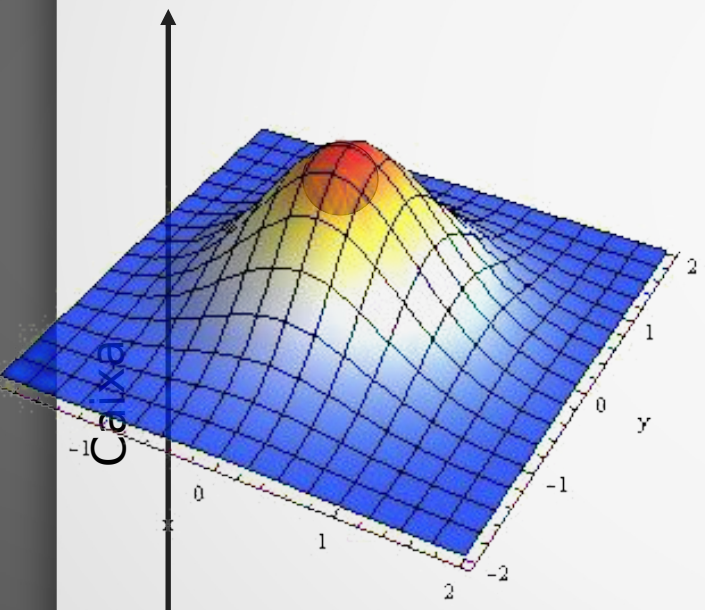


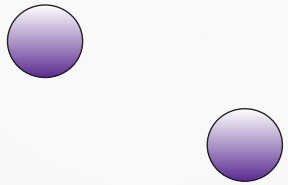
Caixa

Avaliação de usuários IMDB

Exemplo uso de Kernel Gaussiano

- Vencedores do Palme d'Or do festival de Cannes





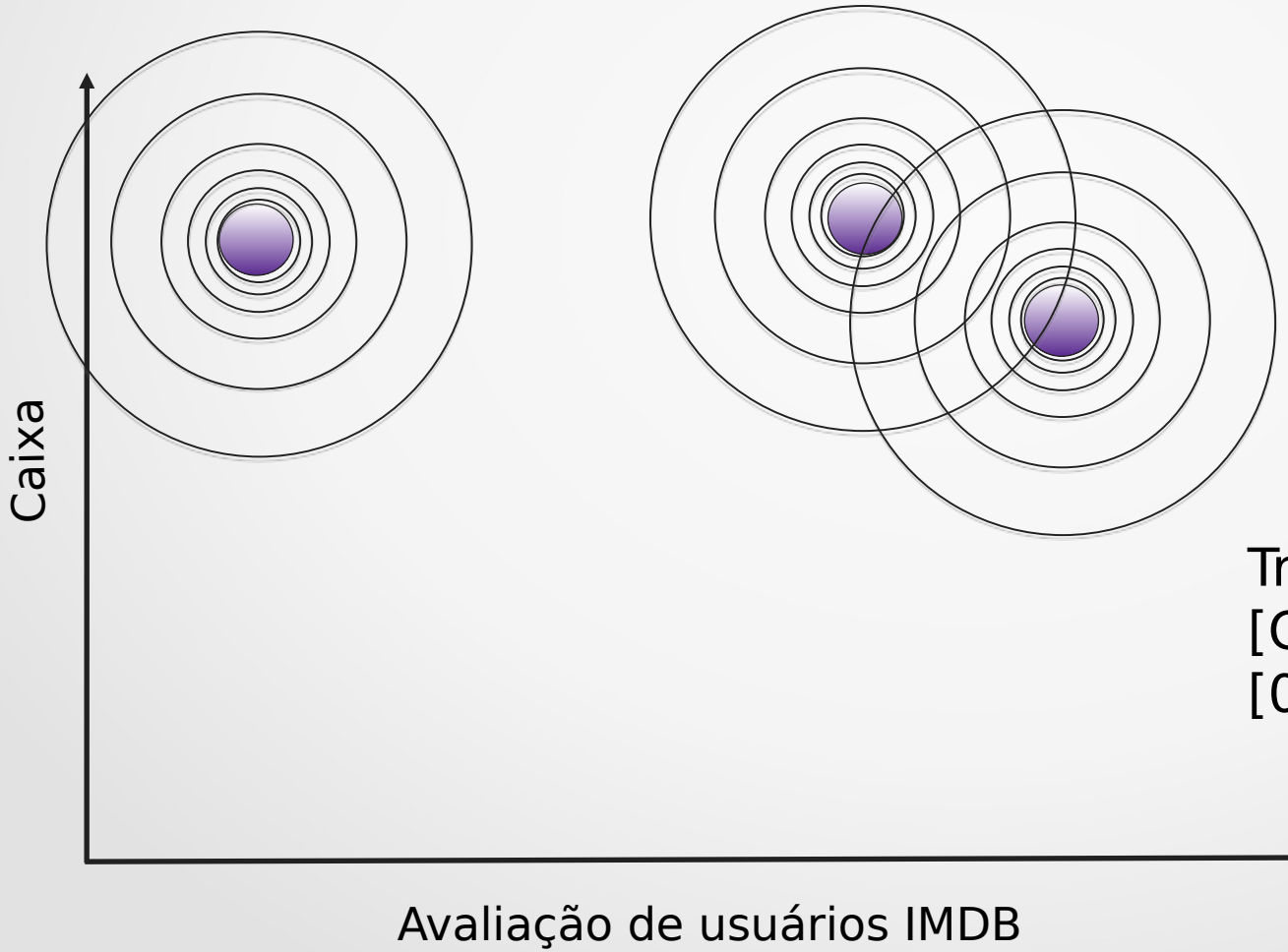
Criar um função gaussiana sobre o novo atributo 3

$$a_1(x^{obs}) = exp\left[\frac{-\sum (x_i^{obs} - x_i^{Transformers})^2}{2\sigma^2}\right]$$

Avaliação de usuários IMDB

Exemplo uso de Kernel Gaussiano

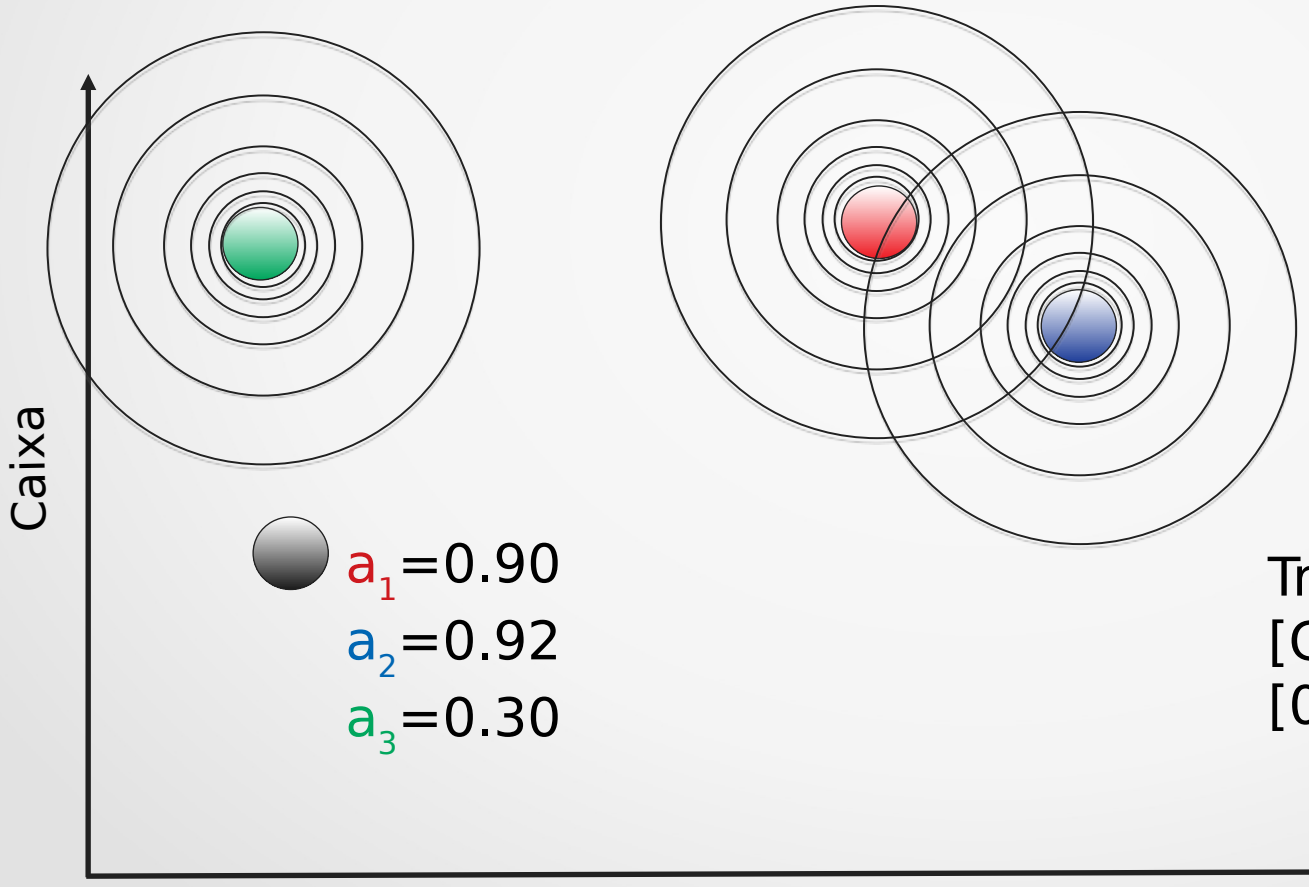
- Vencedores do Palme d'Or do festival de Cannes



Transformação:
[Caixa,Avaliação]ϕ
[0.7a_1, 0.9a_2, 0.6a_3]

Exemplo uso de Kernel Gaussiano

- Vencedores do Palme d'Or do festival de Cannes

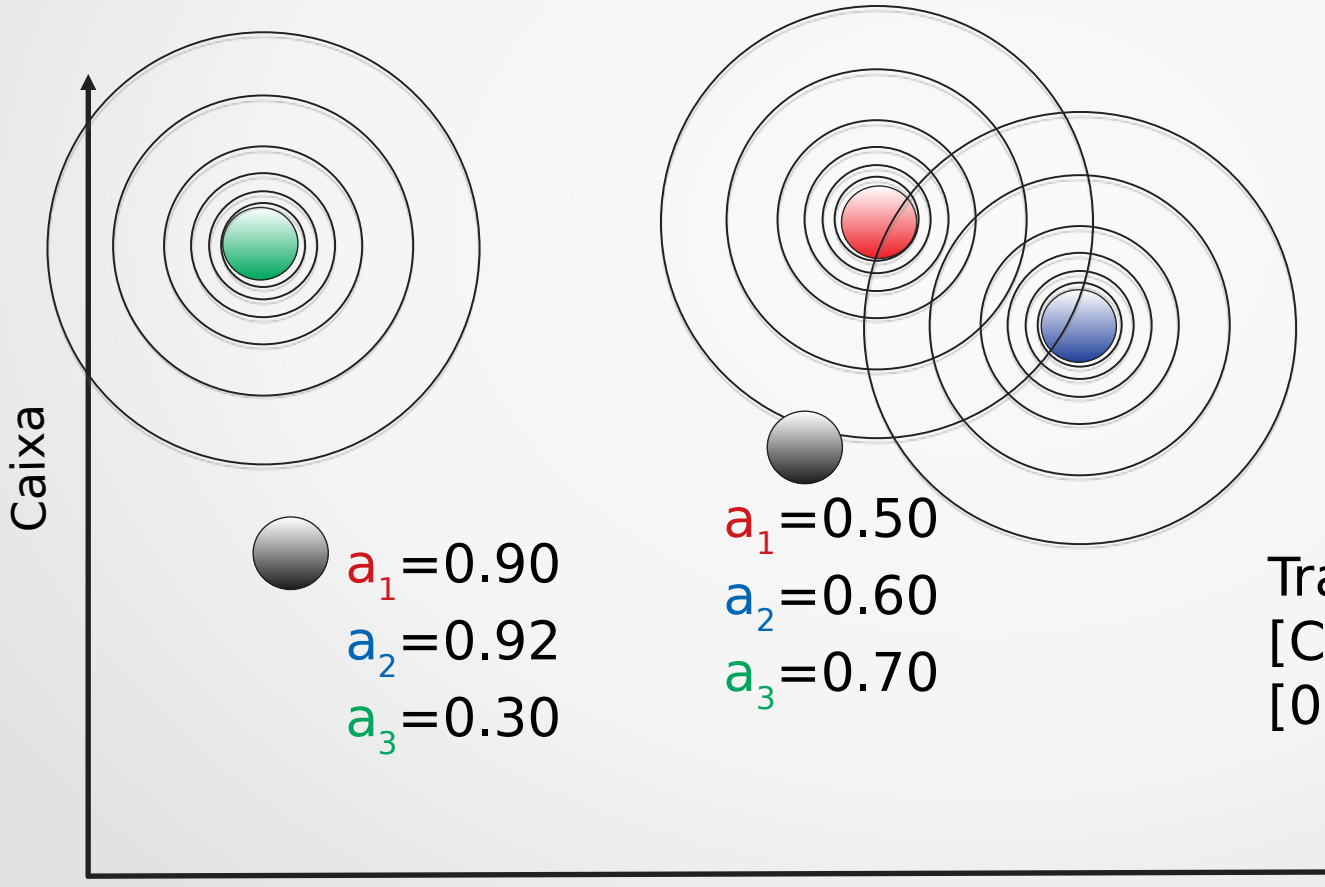


a_1=0.90
a_2=0.92
a_3=0.30

Transformação:
[Caixa,Avaliação]ϕ
[0.7a_1, 0.9a_2, 0.6a_3]

Exemplo uso de Kernel Gaussiano

- Vencedores do Palme d'Or do festival de Cannes



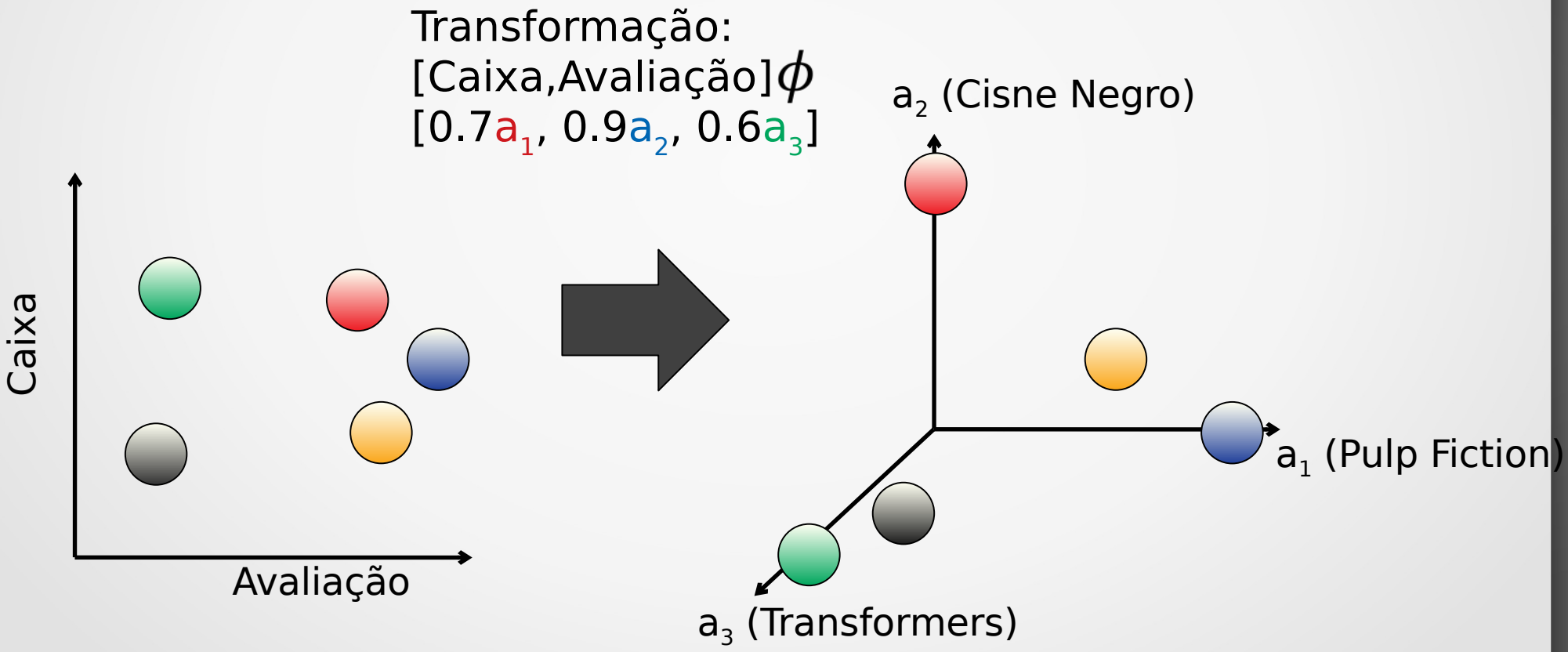
a_1=0.90
a_2=0.92
a_3=0.30

a_1=0.50
a_2=0.60
a_3=0.70

Transformação:
[Caixa,Avaliação]ϕ
[0.7a_1, 0.9a_2, 0.6a_3]

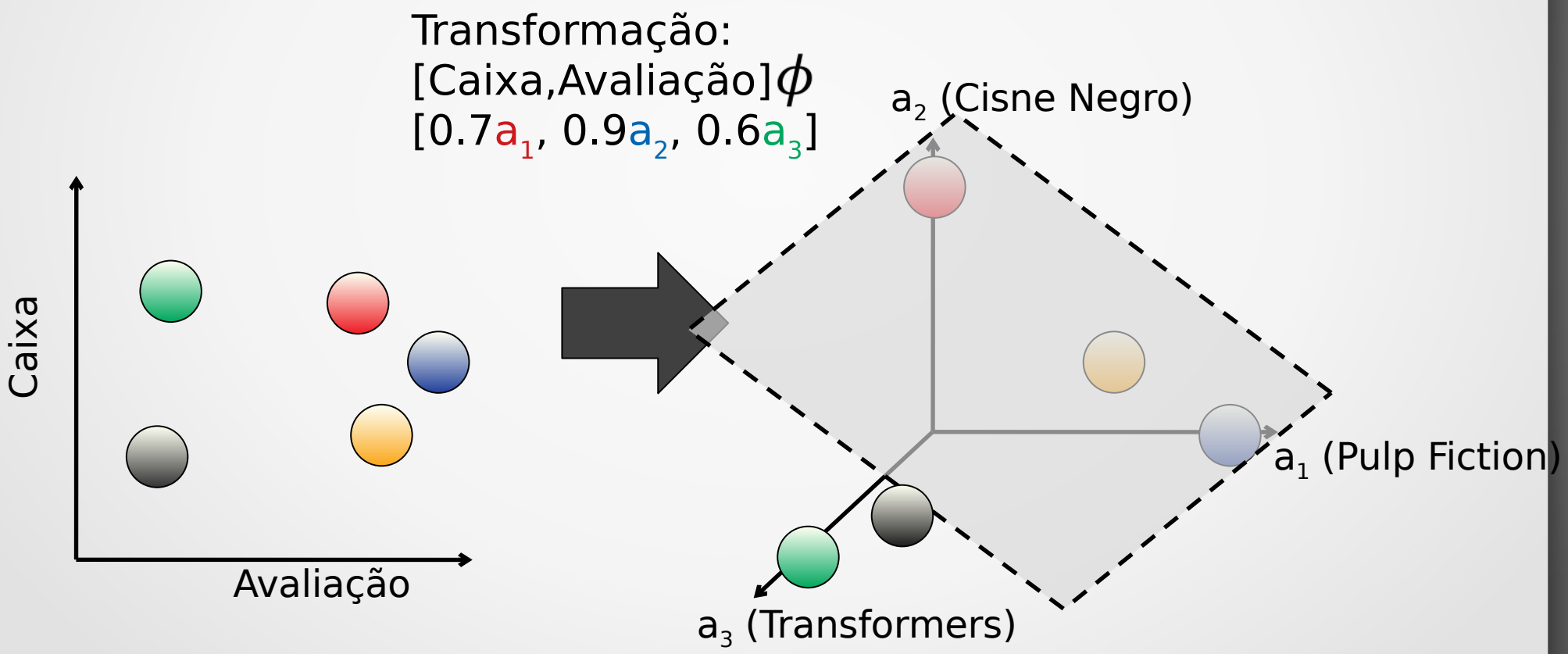
SVM com transformação *Kernel* Gaussiano

- Truque de *kernel* aplicado para modificar espaço



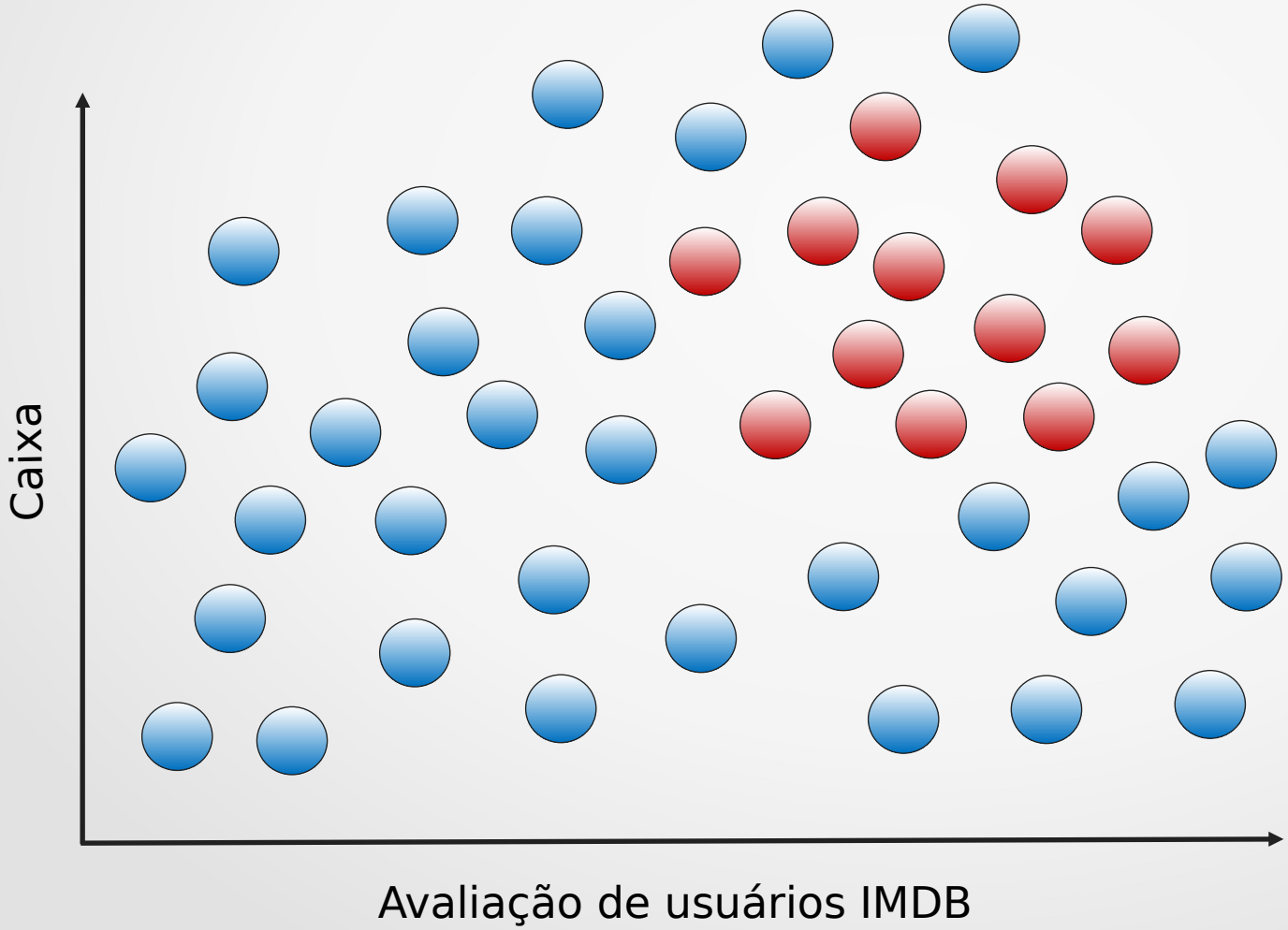
SVM com transformação *Kernel* Gaussiano

- Aplicando hiperplano SVM



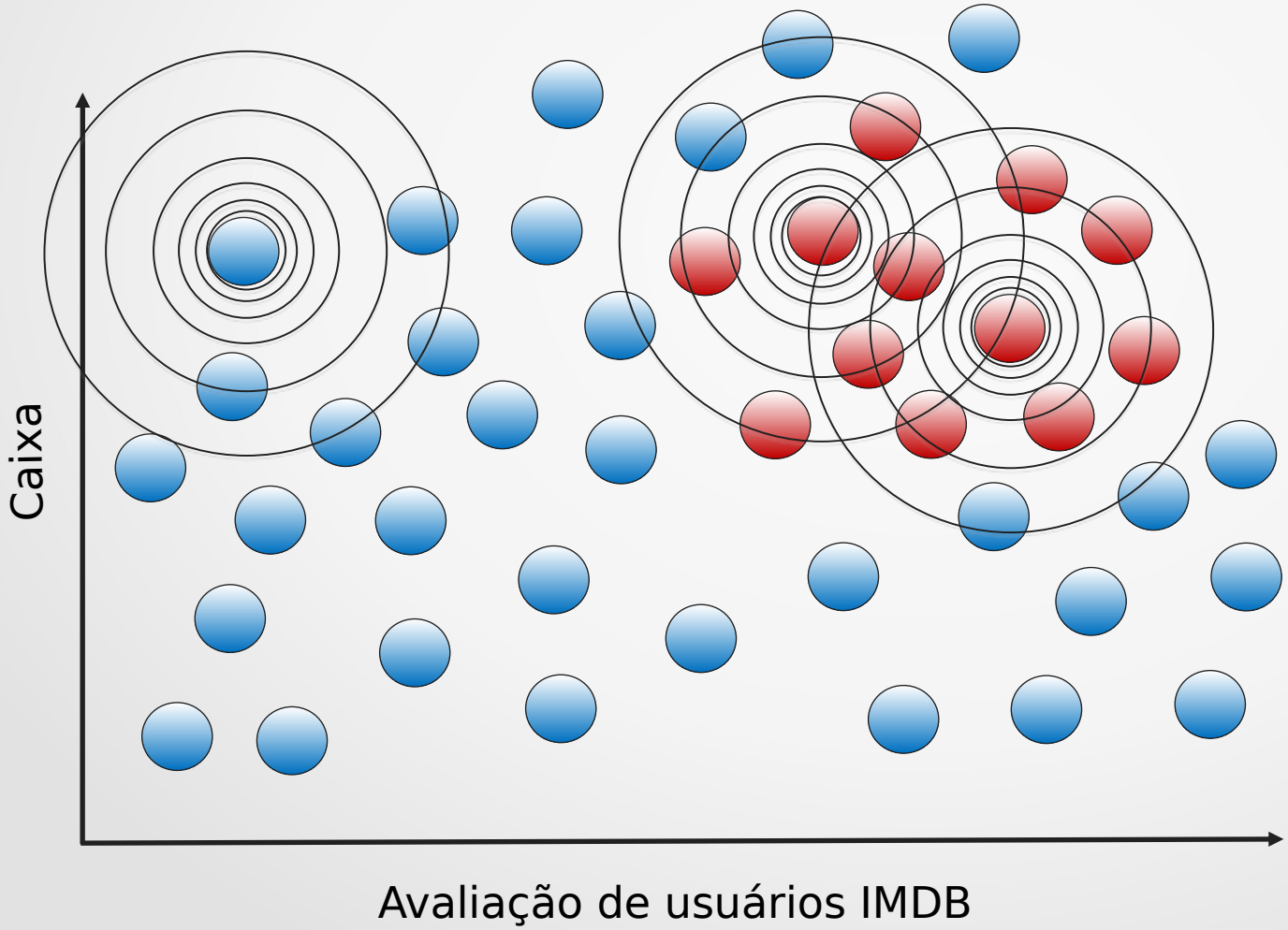
Exemplo uso de Kernel Gaussiano

- Vencedores do Palme d'Or do festival de Cannes



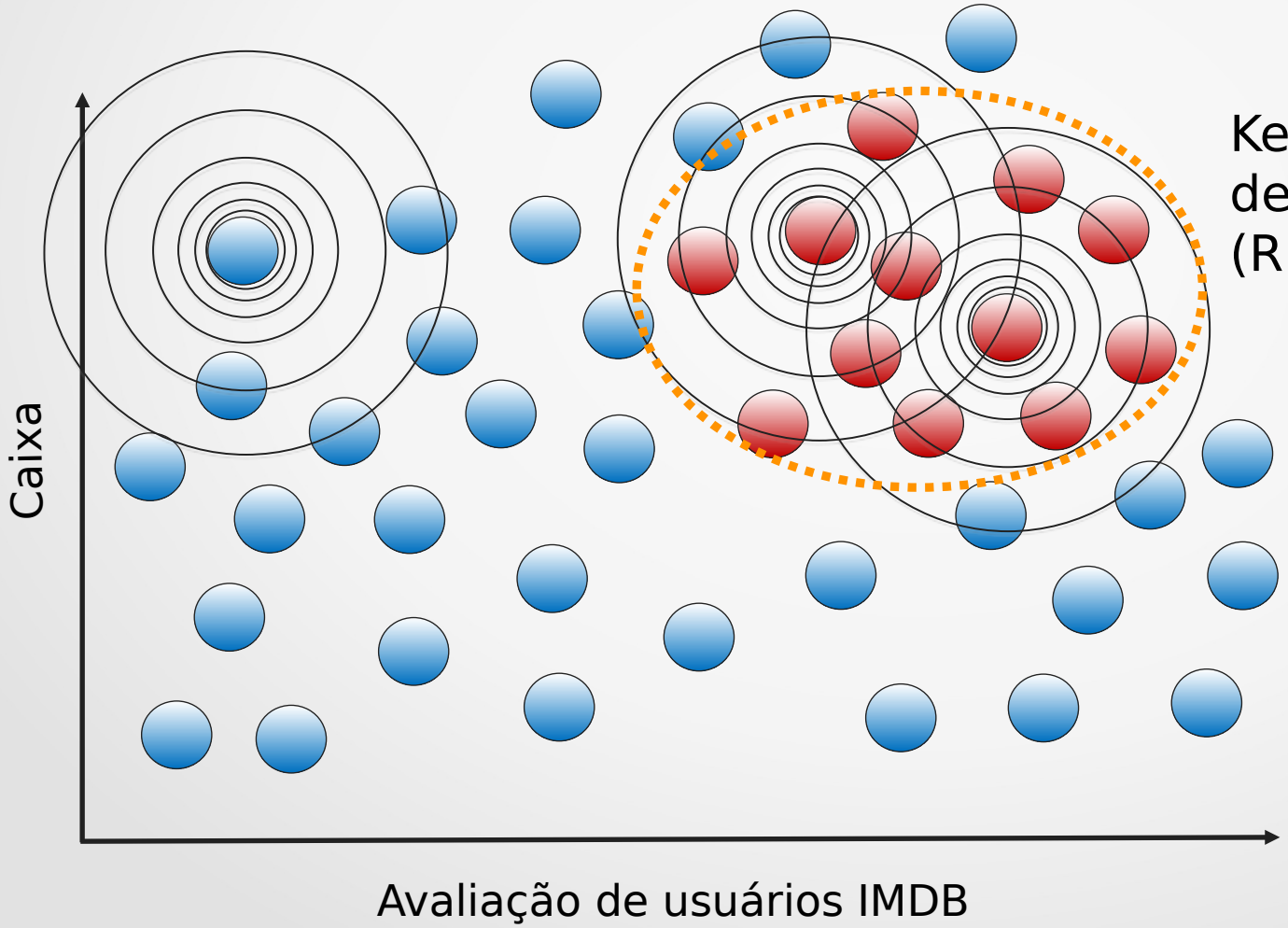
Exemplo uso de Kernel Gaussiano

- Vencedores do Palme d'Or do festival de Cannes



Exemplo uso de Kernel Gaussiano

- Vencedores do Palme d'Or do festival de Cannes



Kernel de Função de Base Radial (RBF)

Aplicando SVM RBF com holdout Iris

```
#Importa SVM do Sklearn
from sklearn.svm import SVC
#Cria uma instância de classe com kernel RBF embutido
rbfSVC = SVC(kernel='rbf', gamma=1.0, C=10.0)


#Ajusta o modelo SVM aos dados de treino
rbfSVC  = rbfSVC .fit(traindata.iloc[:,0:4], traindata.iloc[:,4])


#Classe real
print(testdata.iloc[:,4])
#Classe predita
print(rbfSVC.predict(testdata.iloc[:,0:4]))
```

Aplicando SVM RBF com holdout Iris

- Saída = Classes

```
132      Iris-virginica
50     Iris-versicolor
6          Iris-setosa
113      Iris-virginica
13         Iris-setosa
30         Iris-setosa
18         Iris-setosa
61     Iris-versicolor
104      Iris-virginica
62     Iris-versicolor
87     Iris-versicolor
123      Iris-virginica
149      Iris-virginica
28         Iris-setosa
140      Iris-virginica
```

- Saída Predita

```
['Iris-virginica'
'Iris-versicolor'
'Iris-setosa'
'Iris-virginica'
 'Iris-setosa'
'Iris-setosa'
'Iris-setosa'
'Iris-versicolor'
 'Iris-virginica'
'Iris-versicolor'
'Iris-versicolor'
'Iris-virginica'
 'Iris-virginica'
'Iris-setosa'
'Iris-virginica']
```