

Introdução ao Aprendizado de Máquina

Lucas Gonçalves de Moura Leite

Aula anterior

- ▶ Overfitting
- ▶ Modelos lineares para classificação e regressão
 - ▶ Regressão linear (Regularização)
 - ▶ Regressão polinomial
 - ▶ Regressão logística
 - ▶ SVM linear



Aula de hoje

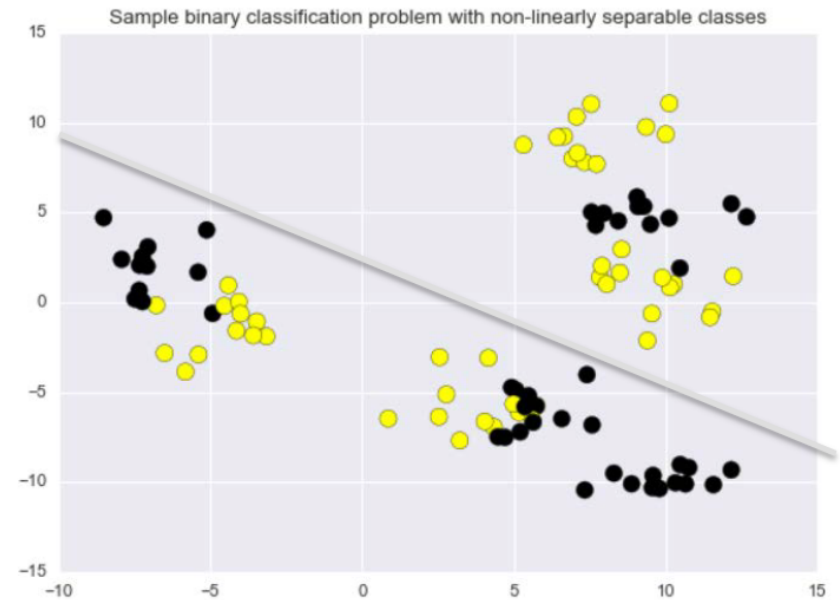
- ▶ SVM não linear
- ▶ Técnicas de validação
- ▶ Árvore de decisão
- ▶ Técnicas de pré-processamento dos atributos



SVM com Kernels

SVM com Kernels

- Classificação para dados não linearmente separáveis

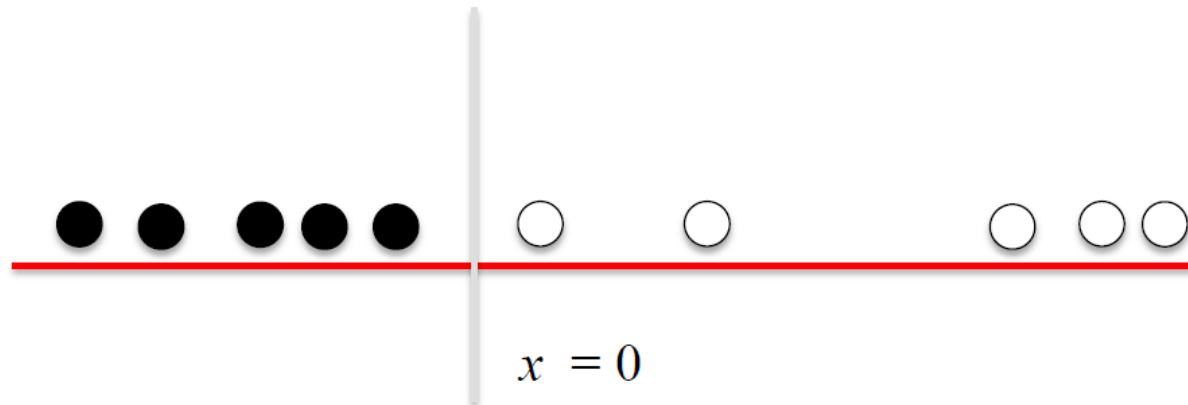


Kernels

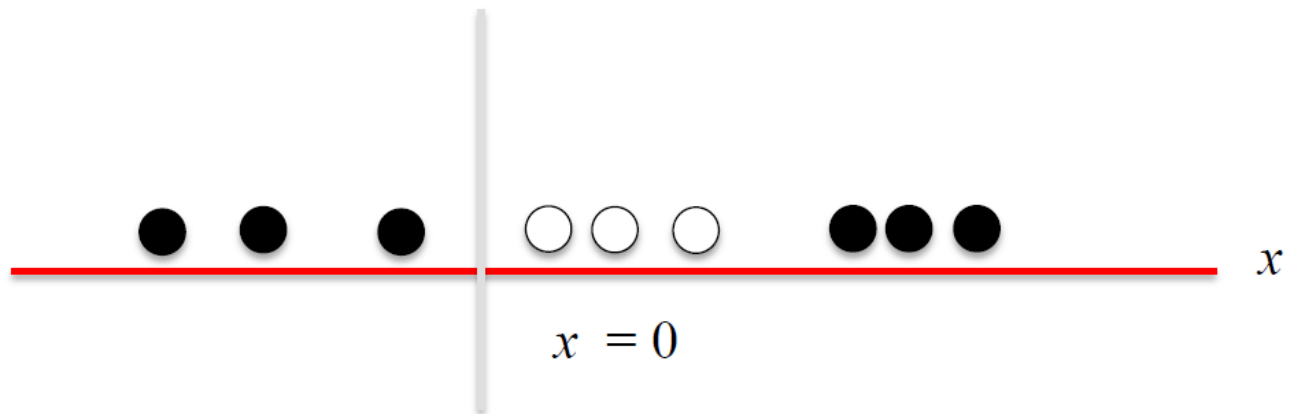
- ▶ Funções que levam o dado da dimensão original para uma nova dimensão através de uma projeção não linear.
- ▶ Maior probabilidade dos dados serem linearmente separáveis nesse novo espaço



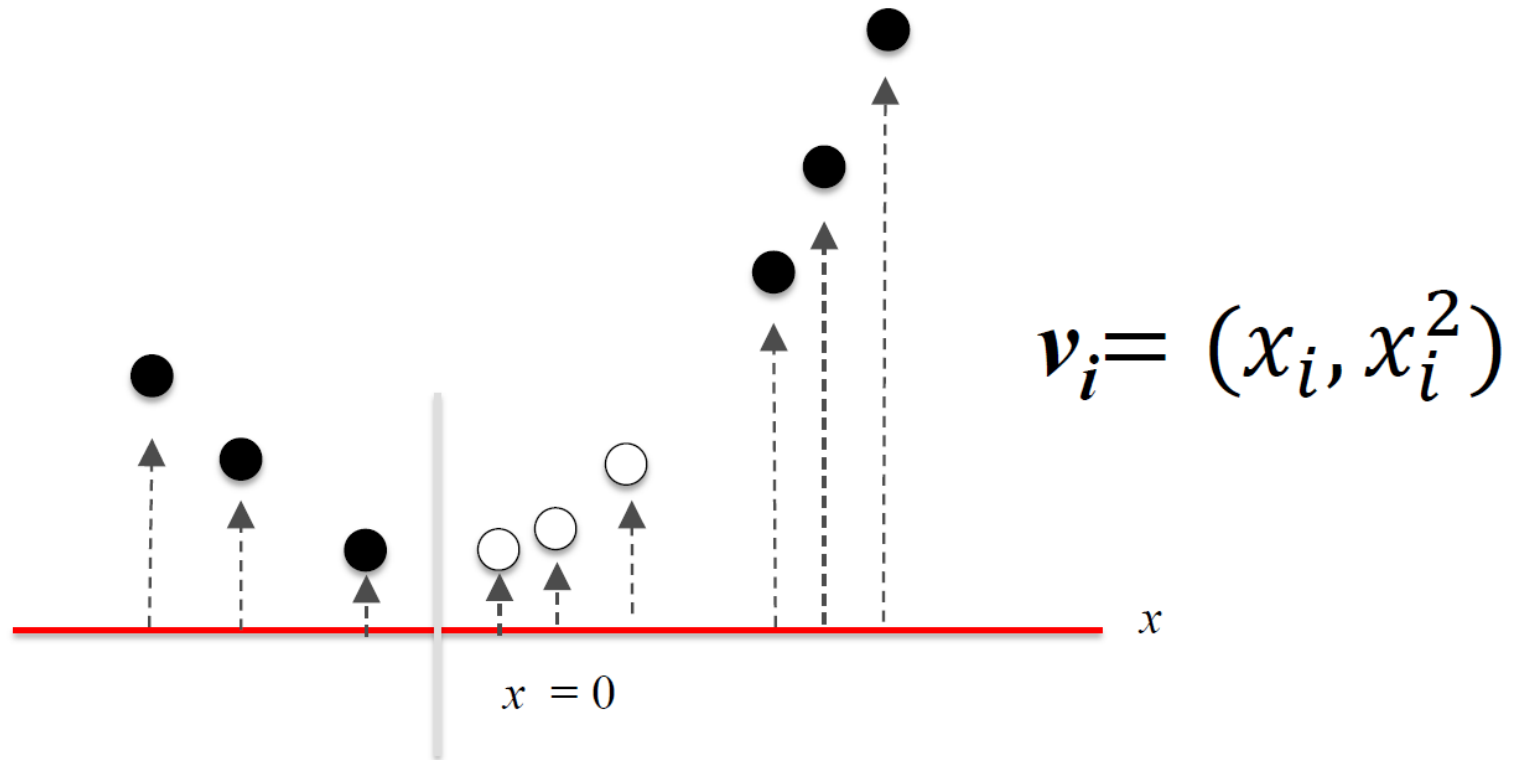
Kernels



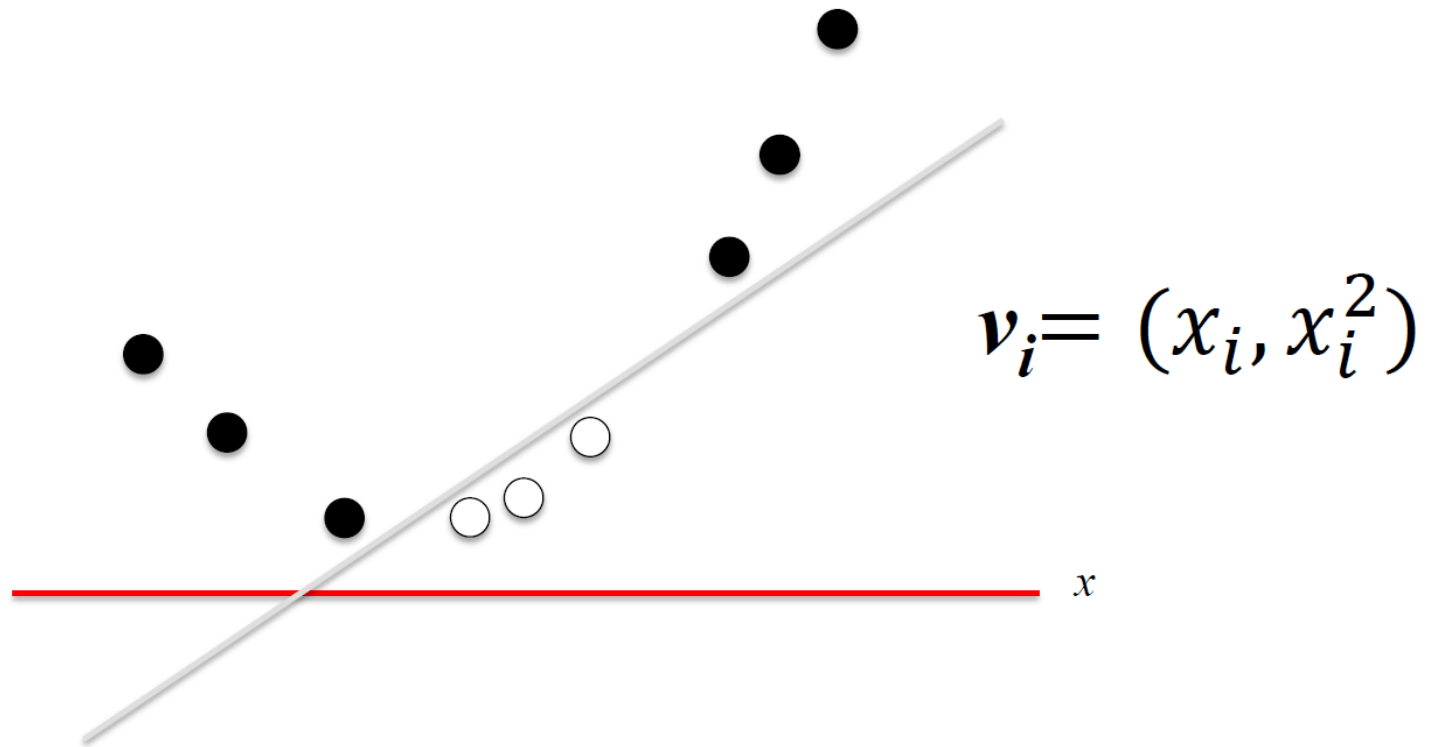
Kernels



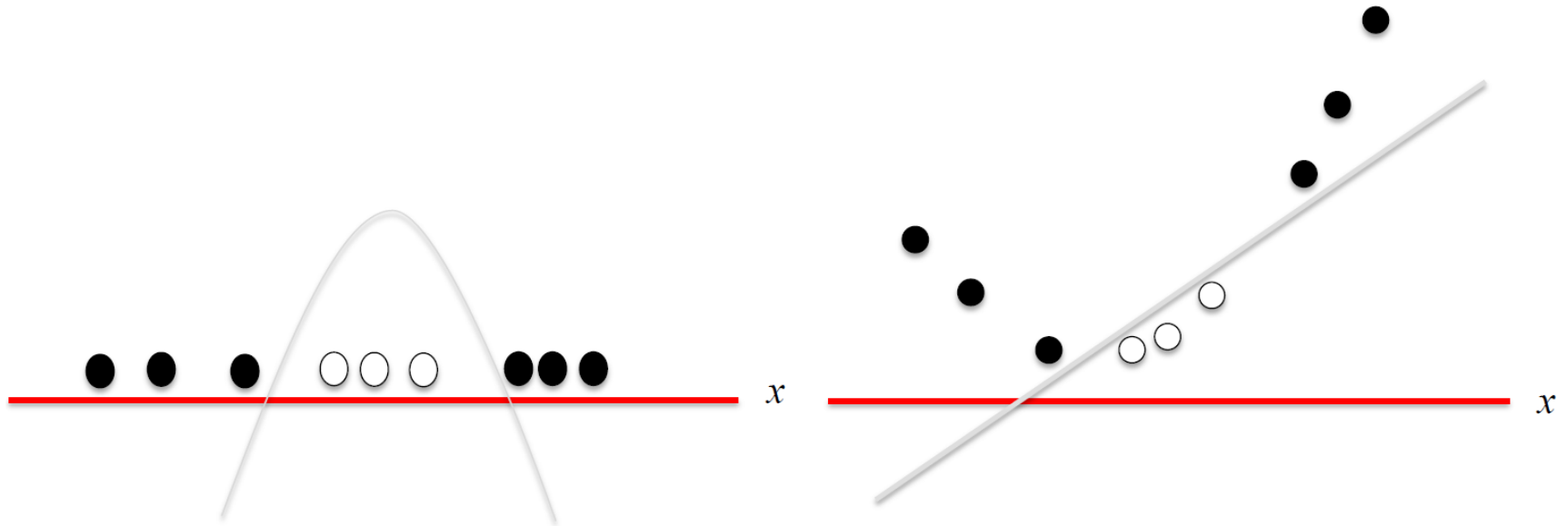
Kernels



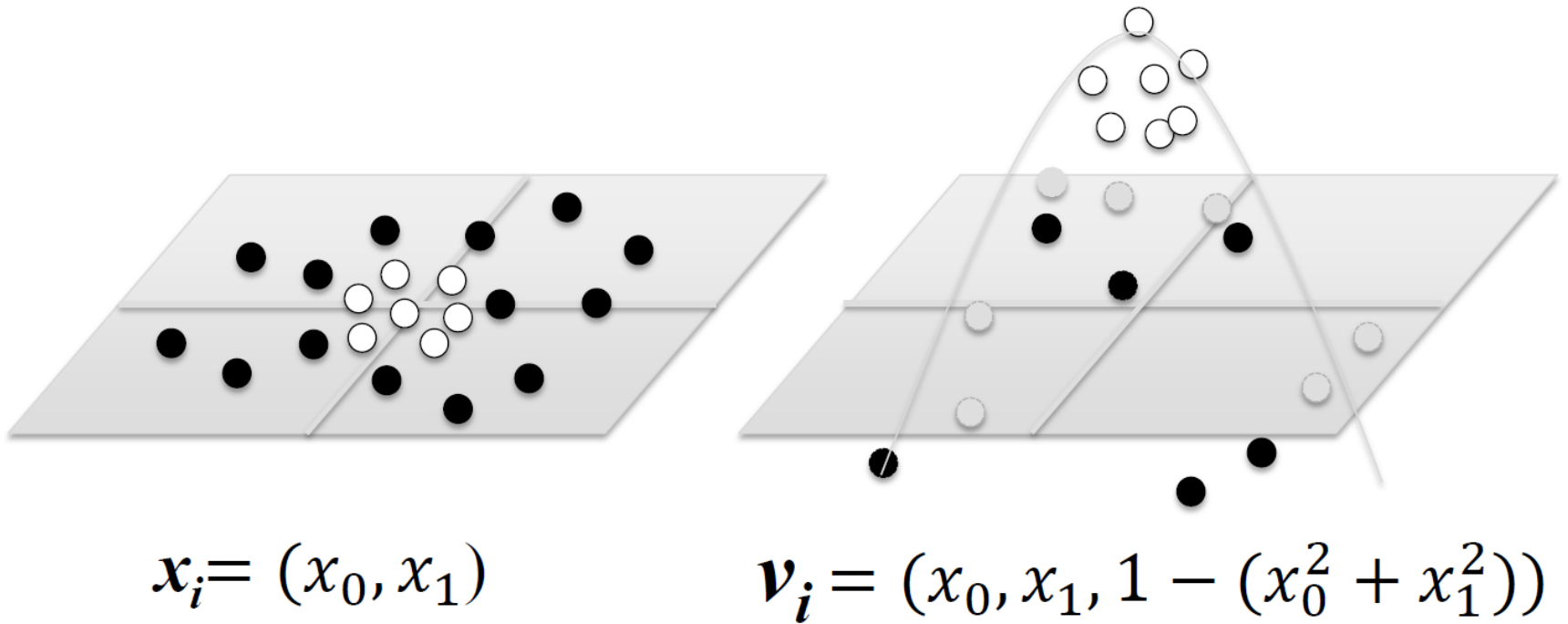
Kernels



Kernels

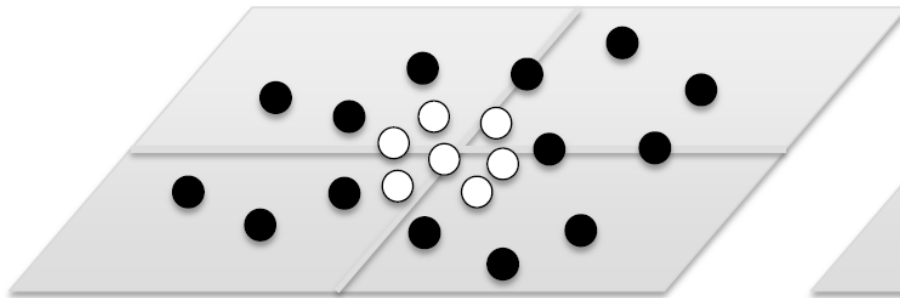


Kernels – Exemplo 2D

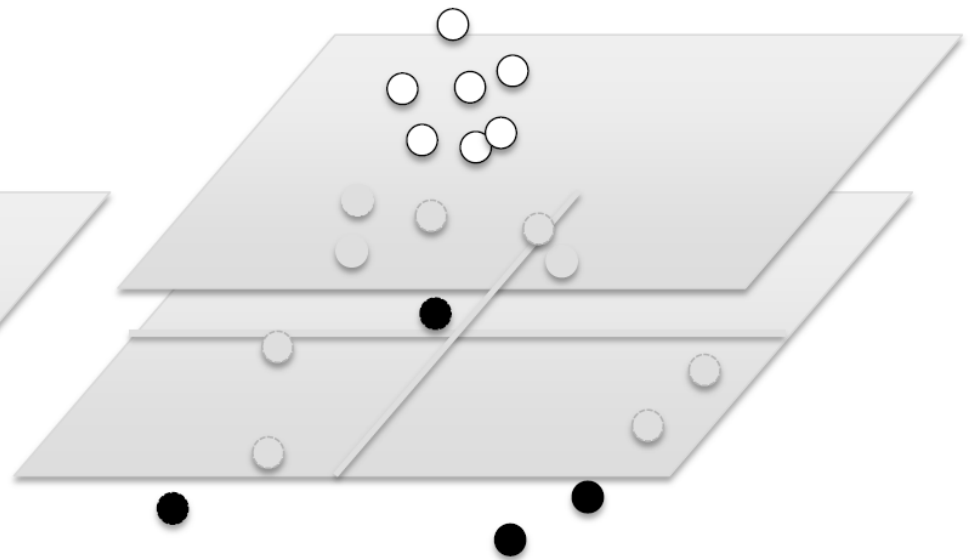


Kernels – Exemplo 2D

Original input space



$$\mathbf{x}_i = (x_0, x_1)$$



$$\mathbf{v}_i = (x_0, x_1, 1 - (x_0^2 + x_1^2))$$



Kernels

► Funções de Kernel

$$K(\vec{x}_i, \vec{x}_j) = \vec{x}_i \cdot \vec{x}_j$$

Linear kernel

$$K(\vec{x}_i, \vec{x}_j) = \exp(-\gamma \|\vec{x}_i - \vec{x}_j\|^2)$$

Gaussian kernel

$$K(\vec{x}_i, \vec{x}_j) = \exp(-\gamma \|\vec{x}_i - \vec{x}_j\|)$$

Exponential kernel

$$K(\vec{x}_i, \vec{x}_j) = (p + \vec{x}_i \cdot \vec{x}_j)^q$$

Polynomial kernel

$$K(\vec{x}_i, \vec{x}_j) = (p + \vec{x}_i \cdot \vec{x}_j)^q \exp(-\gamma \|\vec{x}_i - \vec{x}_j\|^2)$$

Hybrid kernel

$$K(\vec{x}_i, \vec{x}_j) = \tanh(k\vec{x}_i \cdot \vec{x}_j - \delta)$$

Sigmoidal



Kernel trick

- ▶ Dimensão não é aumentada explicitamente
 - ▶ Diferente da regressão polinomial
- ▶ Mais eficiente



Efeitos dos Parâmetros

- ▶ SVM

- ▶ C

- ▶ Permitir erro

- ▶ Parâmetro do Kernel

- ▶ Gamma (RBF), Degree(Poly)



Efeitos dos Parâmetros

- ▶ **Gamma (RBF)**
 - ▶ Aumentar -> superfícies mais complexas



SVM com Kernel

- ▶ **Vantagens**

- ▶ Funcionam bem para uma vasta gama de problemas
- ▶ Funcionam bem para dados com muitos ou poucos atributos

- ▶ **Desvantagens**

- ▶ Baixa interpretabilidade
- ▶ Custo computacional



Exercício

- ▶ Carregar os dados do dataset breast cancer
- ▶ Classificar usando o SVM com Kernel

```
from sklearn.datasets import load_breast_cancer  
cancer = load_breast_cancer()  
(X_cancer, y_cancer) = load_breast_cancer(return_X_y = True)
```





Validação

Aprendizado de Máquina

- ▶ Métodos Supervisionados
- ▶ Procedimento
 - ▶ Dividir em treino e teste
 - ▶ Treinar o modelo com dados de treino (.fit)
 - ▶ Aplicar o método para dados não vistos (.predict ou .score)




Divisão Treino e Teste

- ▶ Avaliar performance em dados que não foram vistos
- ▶ Uma só divisão
 - ▶ Pode ter “sorte” e encontrar uma boa divisão
- ▶ Validação cruzada
 - ▶ Múltiplas divisões
 - ▶ Repetir o processo



Validação Cruzada

Original dataset		Model 1	Model 2	Model 3	Model 4	Model 5
	Fold 1	Test	Train	Train	Train	Train
	Fold 2	Train	Test	Train	Train	Train
	Fold 3	Train	Train	Test	Train	Train
	Fold 4	Train	Train	Train	Test	Train
	Fold 5	Train	Train	Train	Train	Test



Validação Cruzada

► Scikit-learn

```
import numpy as np
import pandas as pd
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier

df = pd.read_csv('dadosClassBin.csv', index_col=0)

X = df[['x1', 'x2']]
y = df['y']

clf = KNeighborsClassifier(n_neighbors = 5)

cv_scores = cross_val_score(clf, X, y)

print('Cross-validation scores (3-fold):', cv_scores)
print('Mean cross-validation score (3-fold): {:.3f}'
      .format(np.mean(cv_scores)))
```

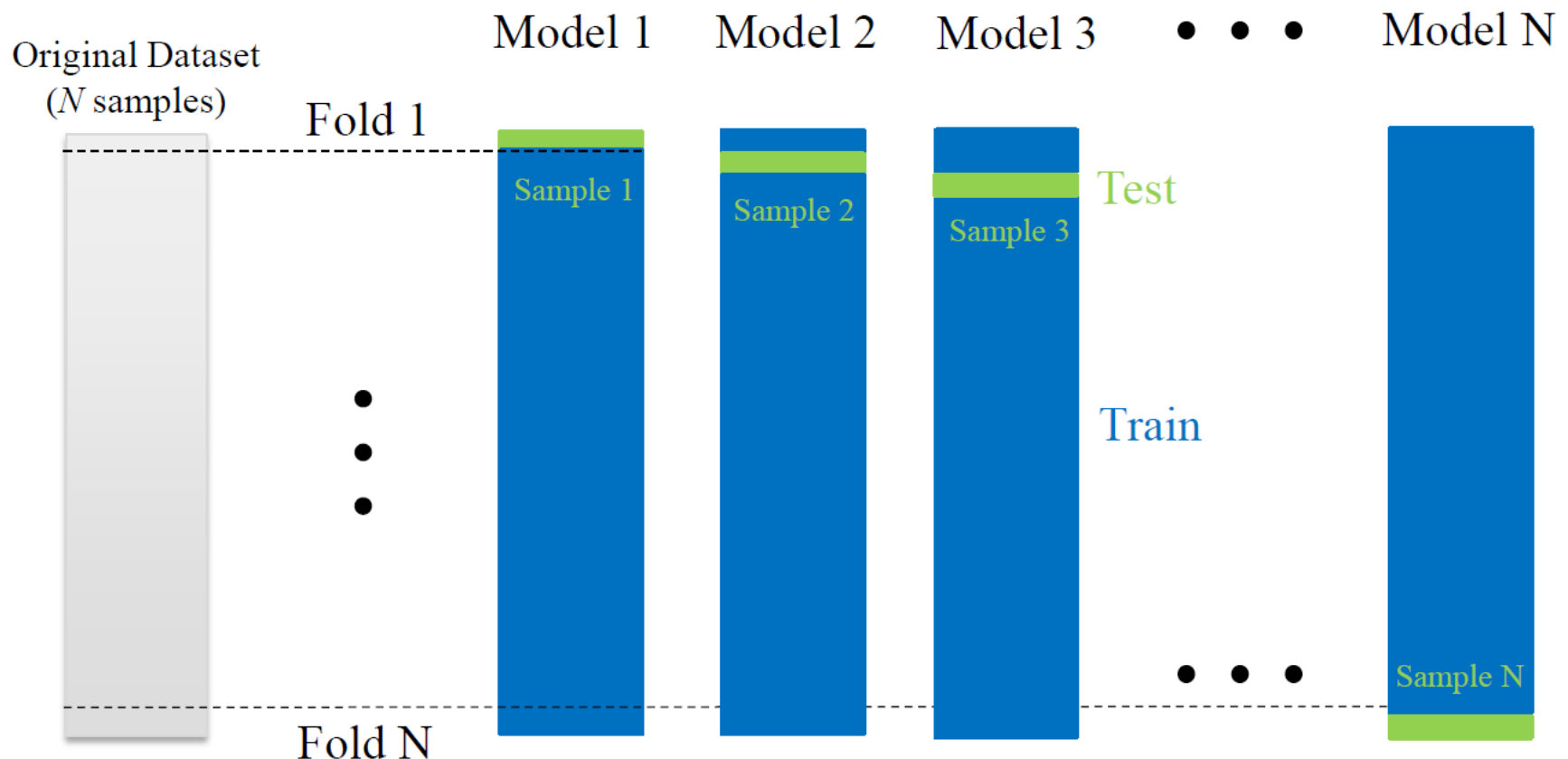

Validação Cruzada Estratificada

- ▶ Mantem proporção das classes
- ▶ Ordem dos dados é escolhida aleatoriamente
- ▶ Usado no scikit learn para classificação
- ▶ Regressão
 - ▶ Validação cruzada convencional



Validação Leave-one-out

- ▶ K é igual ao número de dados
- ▶ Poucos dados



Curva de Validação

- ▶ Variar parâmetros e observar performance do método

```
from sklearn.svm import SVC
from sklearn.model_selection import validation_curve

param_range = np.linspace(1, 3, num=5)
train_scores, test_scores = validation_curve(SVC(), X, y,
                                             param_name='gamma',
                                             param_range=param_range, cv=3)
```



Árvore de Decisão

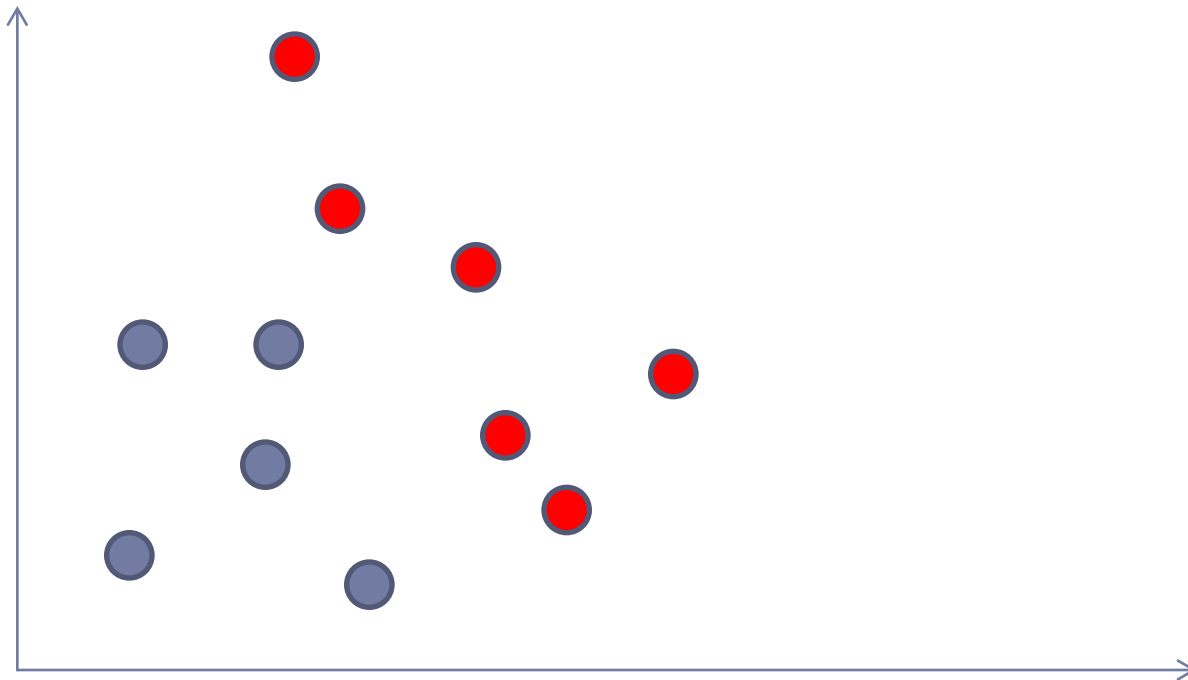
Árvore de Decisão

- ▶ Conjunto de regras se-então utilizadas para classificação e regressão



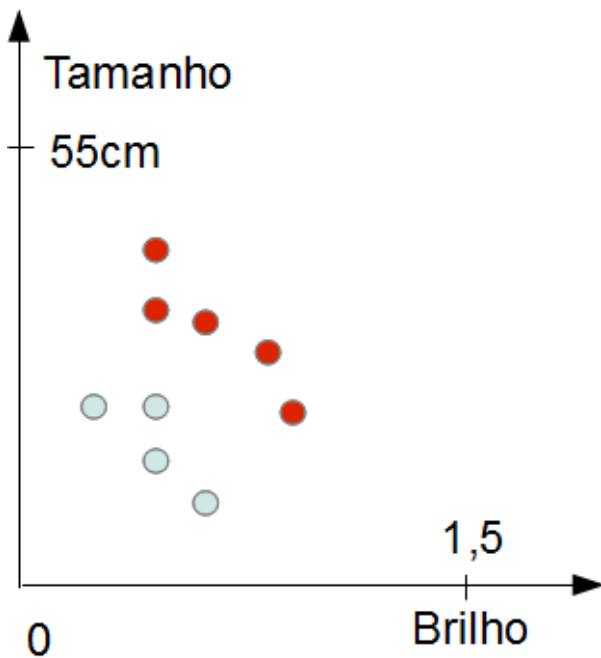
Árvore de Decisão

- ▶ Conjunto de regras se-então utilizadas para classificação e regressão



Árvore de Decisão

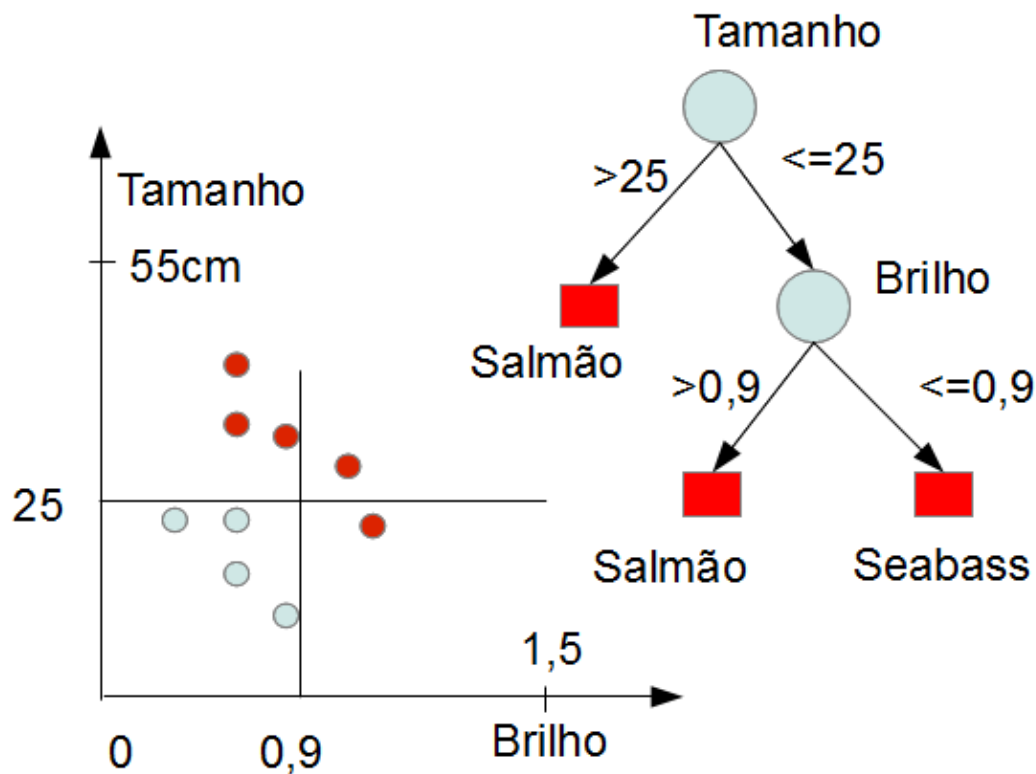
► Exemplo



Brilho	Tamanho	Classificação
1,2	23	Salmão
1,1	30	Salmão
0,9	36	Salmão
0,8	45	Salmão
0,8	38	Salmão
0,9	15	Seabass
0,8	20	Seabass
0,8	25	Seabass
0,7	25	Seabass

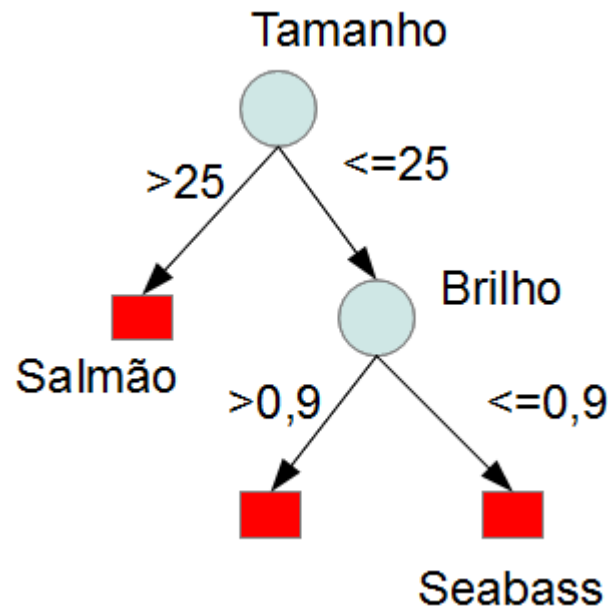
Árvore de Decisão

► Exemplo



Árvore de Decisão

- ▶ Como obter a árvore, automaticamente, a partir dos dados ?



Árvore de Decisão

- ▶ Como obter a árvore, automaticamente, a partir dos dados ?

Brilho	Tamanho	Classificação
1,2	23	Salmão
1,1	30	Salmão
0,9	36	Salmão
0,8	45	Salmão
0,8	38	Salmão
0,9	15	Seabass
0,8	20	Seabass
0,8	25	Seabass
0,7	25	Seabass



Árvore de Decisão

- ▶ Como obter a árvore, automaticamente, a partir dos dados ?
- ▶ Utilizar um índice de impureza do resultado
- ▶ Vários índices (entropia, gini ...)

Brilho	Tamanho	Classificação
1,2	23	Salmão
1,1	30	Salmão
0,9	36	Salmão
0,8	45	Salmão
0,8	38	Salmão
0,9	15	Seabass
0,8	20	Seabass
0,8	25	Seabass
0,7	25	Seabass



Árvore de Decisão

- ▶ Gini

- ▶ $G = 1 - \sum_{i=1}^m f_i^2$



Árvore de Decisão

► Gini

► $G = 1 - \sum_{i=1}^m f_i^2$

► Exemplo

► 4 Seabass e 5 salmão

Brilho	Tamanho	Classificação
1,2	23	Salmão
1,1	30	Salmão
0,9	36	Salmão
0,8	45	Salmão
0,8	38	Salmão
0,9	15	Seabass
0,8	20	Seabass
0,8	25	Seabass
0,7	25	Seabass

Árvore de Decisão

▶ Gini

$$▶ G = 1 - \sum_{i=1}^m f_i^2$$

▶ Exemplo

▶ 4 Seabass e 5 salmão

$$▶ G = 1 - \left[\left(\frac{4}{9} \right)^2 + \left(\frac{5}{9} \right)^2 \right] = 0.5$$

Brilho	Tamanho	Classificação
1,2	23	Salmão
1,1	30	Salmão
0,9	36	Salmão
0,8	45	Salmão
0,8	38	Salmão
0,9	15	Seabass
0,8	20	Seabass
0,8	25	Seabass
0,7	25	Seabass

Árvore de Decisão

▶ Gini

▶ $G = 1 - \sum_{i=1}^m f_i^2$

▶ Exemplo

▶ Escolhendo $B > 0.7$

Brilho	Tamanho	Classificação
1,2	23	Salmão
1,1	30	Salmão
0,9	36	Salmão
0,8	45	Salmão
0,8	38	Salmão
0,9	15	Seabass
0,8	20	Seabass
0,8	25	Seabass
0,7	25	Seabass

Árvore de Decisão

▶ Gini

$$▶ G = 1 - \sum_{i=1}^m f_i^2$$

▶ Exemplo

- ▶ Escolhendo $B > 0.7$
- ▶ Dois lados
 - ▶ 1 Seabass e 0 salmão
 - ▶ 3 Seabass e 5 salmão

Brilho	Tamanho	Classificação
1,2	23	Salmão
1,1	30	Salmão
0,9	36	Salmão
0,8	45	Salmão
0,8	38	Salmão
0,9	15	Seabass
0,8	20	Seabass
0,8	25	Seabass
0,7	25	Seabass

Árvore de Decisão

▶ Gini

▶ $G = 1 - \sum_{i=1}^m f_i^2$

▶ Exemplo

▶ Escolhendo $B > 0.7$

▶ Dois lados

▶ 1 Seabass e 0 salmão

□ $G = 1 - \left[\left(\frac{1}{1} \right)^2 + \left(\frac{0}{1} \right)^2 \right]$

▶ 3 Seabass e 5 salmão

□ $G = 1 - \left[\left(\frac{3}{8} \right)^2 + \left(\frac{5}{8} \right)^2 \right]$

Brilho	Tamanho	Classificação
1,2	23	Salmão
1,1	30	Salmão
0,9	36	Salmão
0,8	45	Salmão
0,8	38	Salmão
0,9	15	Seabass
0,8	20	Seabass
0,8	25	Seabass
0,7	25	Seabass

Árvore de Decisão

▶ Gini

▶ $G = 1 - \sum_{i=1}^m f_i^2$

▶ Exemplo

▶ Escolhendo $B > 0.7$

▶ Dois lados

▶ 1 Seabass e 0 salmão

□ $G = 1 - [1^2 + 0^2] = 0$

▶ 3 Seabass e 5 salmão

□ $G = 1 - [0.375^2 + 0.625^2] = 0.47$

Brilho	Tamanho	Classificação
1,2	23	Salmão
1,1	30	Salmão
0,9	36	Salmão
0,8	45	Salmão
0,8	38	Salmão
0,9	15	Seabass
0,8	20	Seabass
0,8	25	Seabass
0,7	25	Seabass

Árvore de Decisão

▶ Gini

$$▶ G = 1 - \sum_{i=1}^m f_i^2$$

▶ Exemplo

▶ Escolhendo $B > 0.7$

▶ Dois lados

▶ 1 Seabass e 0 salmão

$$\square G = 1 - [1^2 + 0^2] = 0$$

▶ 3 Seabass e 5 salmão

$$\square G = 1 - [0.375^2 + 0.625^2] = 0.47$$

▶ Total

$$▶ G = 1 - \left[\frac{1}{9} 1 + \frac{8}{9} 0.53 \right] = 0.42$$

Brilho	Tamanho	Classificação
1,2	23	Salmão
1,1	30	Salmão
0,9	36	Salmão
0,8	45	Salmão
0,8	38	Salmão
0,9	15	Seabass
0,8	20	Seabass
0,8	25	Seabass
0,7	25	Seabass

Árvore de Decisão

► Gini

► $G = 1 - \sum_{i=1}^m f_i^2$

► Exemplo

► Escolhendo $T > 25$

Brilho	Tamanho	Classificação
1,2	23	Salmão
1,1	30	Salmão
0,9	36	Salmão
0,8	45	Salmão
0,8	38	Salmão
0,9	15	Seabass
0,8	20	Seabass
0,8	25	Seabass
0,7	25	Seabass

Árvore de Decisão

▶ Gini

$$▶ G = 1 - \sum_{i=1}^m f_i^2$$

▶ Exemplo

▶ Escolhendo $T > 25$

▶ Dois lados

▶ 4 Seabass e 1 salmão

▶ 0 Seabass e 4 salmão

Brilho	Tamanho	Classificação
1,2	23	Salmão
1,1	30	Salmão
0,9	36	Salmão
0,8	45	Salmão
0,8	38	Salmão
0,9	15	Seabass
0,8	20	Seabass
0,8	25	Seabass
0,7	25	Seabass

Árvore de Decisão

▶ Gini

$$▶ G = 1 - \sum_{i=1}^m f_i^2$$

▶ Exemplo

▶ Escolhendo $T > 25$

▶ Dois lados

▶ 4 Seabass e 1 salmão

$$\square G = 1 - \left[\left(\frac{4}{5} \right)^2 + \left(\frac{1}{5} \right)^2 \right]$$

▶ 0 Seabass e 4 salmão

$$\square G = 1 - \left[\left(\frac{0}{4} \right)^2 + \left(\frac{4}{4} \right)^2 \right]$$

Brilho	Tamanho	Classificação
1,2	23	Salmão
1,1	30	Salmão
0,9	36	Salmão
0,8	45	Salmão
0,8	38	Salmão
0,9	15	Seabass
0,8	20	Seabass
0,8	25	Seabass
0,7	25	Seabass

Árvore de Decisão

▶ Gini

▶ $G = 1 - \sum_{i=1}^m f_i^2$

▶ Exemplo

▶ Escolhendo $T > 25$

▶ Dois lados

▶ 4 Seabass e 1 salmão

□ $G = 1 - [0.8^2 + 0.2^2] = 0.32$

▶ 0 Seabass e 4 salmão

□ $G = 1 - [0^2 + 1^2] = 0$

Brilho	Tamanho	Classificação
1,2	23	Salmão
1,1	30	Salmão
0,9	36	Salmão
0,8	45	Salmão
0,8	38	Salmão
0,9	15	Seabass
0,8	20	Seabass
0,8	25	Seabass
0,7	25	Seabass

Árvore de Decisão

▶ Gini

$$▶ G = 1 - \sum_{i=1}^m f_i^2$$

▶ Exemplo

▶ Escolhendo $T > 25$

▶ Dois lados

▶ 4 Seabass e 1 salmão

$$\square G = 1 - [0.8^2 + 0.2^2] = 0.32$$

▶ 0 Seabass e 4 salmão

$$\square G = 1 - [0^2 + 1^2] = 0$$

▶ Total

$$▶ G = 1 - \left[\frac{5}{9} 0.68 + \frac{4}{9} 1 \right] = 0.18$$

Brilho	Tamanho	Classificação
1,2	23	Salmão
1,1	30	Salmão
0,9	36	Salmão
0,8	45	Salmão
0,8	38	Salmão
0,9	15	Seabass
0,8	20	Seabass
0,8	25	Seabass
0,7	25	Seabass

Árvore de Decisão no Scikit-learn

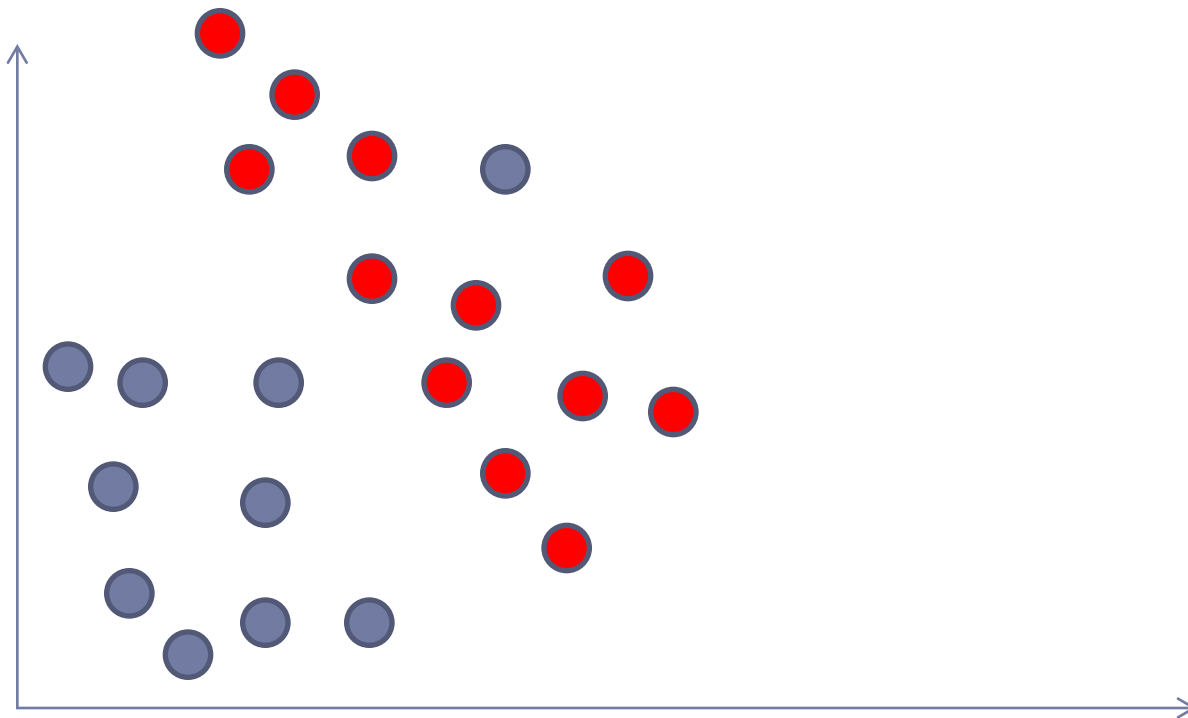
```
from sklearn.datasets import load_iris
from sklearn import tree
from adspy_shared_utilities import plot_decision_tree
from sklearn.model_selection import train_test_split
import graphviz

iris = load_iris()

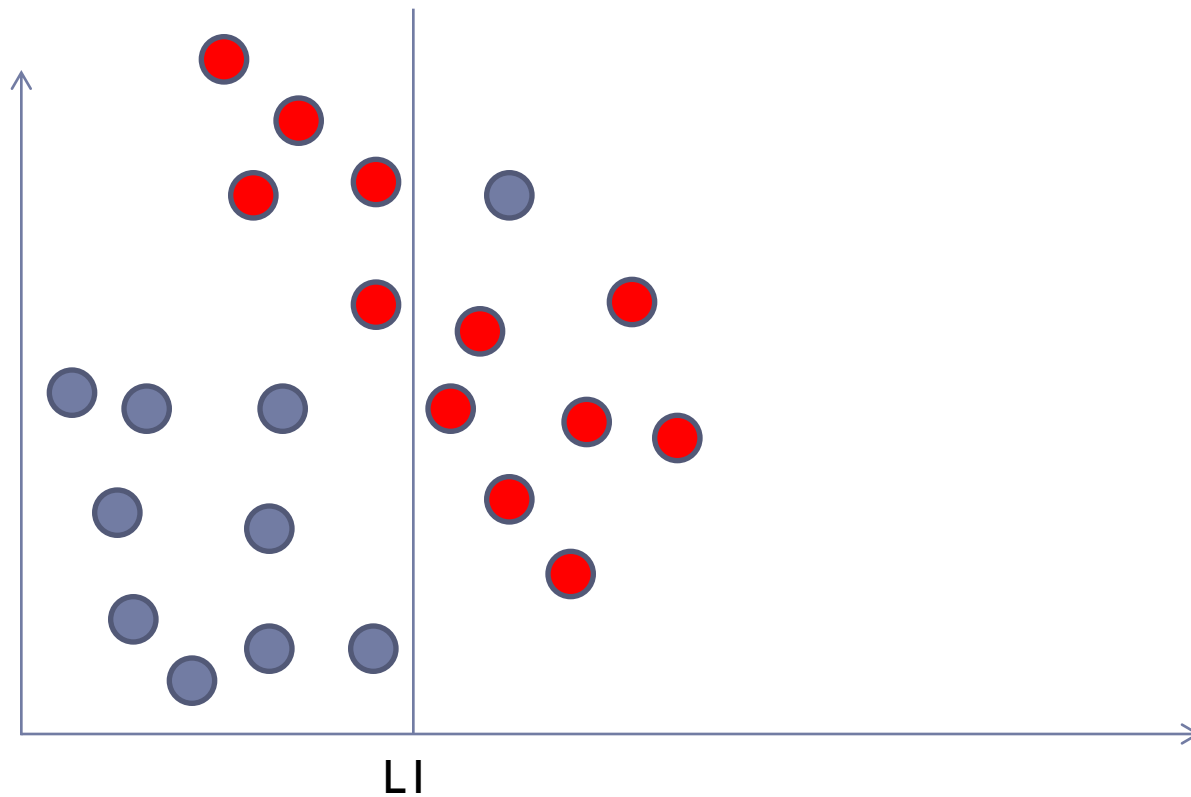
X_train, X_test, y_train, y_test = train_test_split(iris.data, iris.target, random_state = 3)
clf = tree.DecisionTreeClassifier(max_depth = 4).fit(X_train, y_train)
```



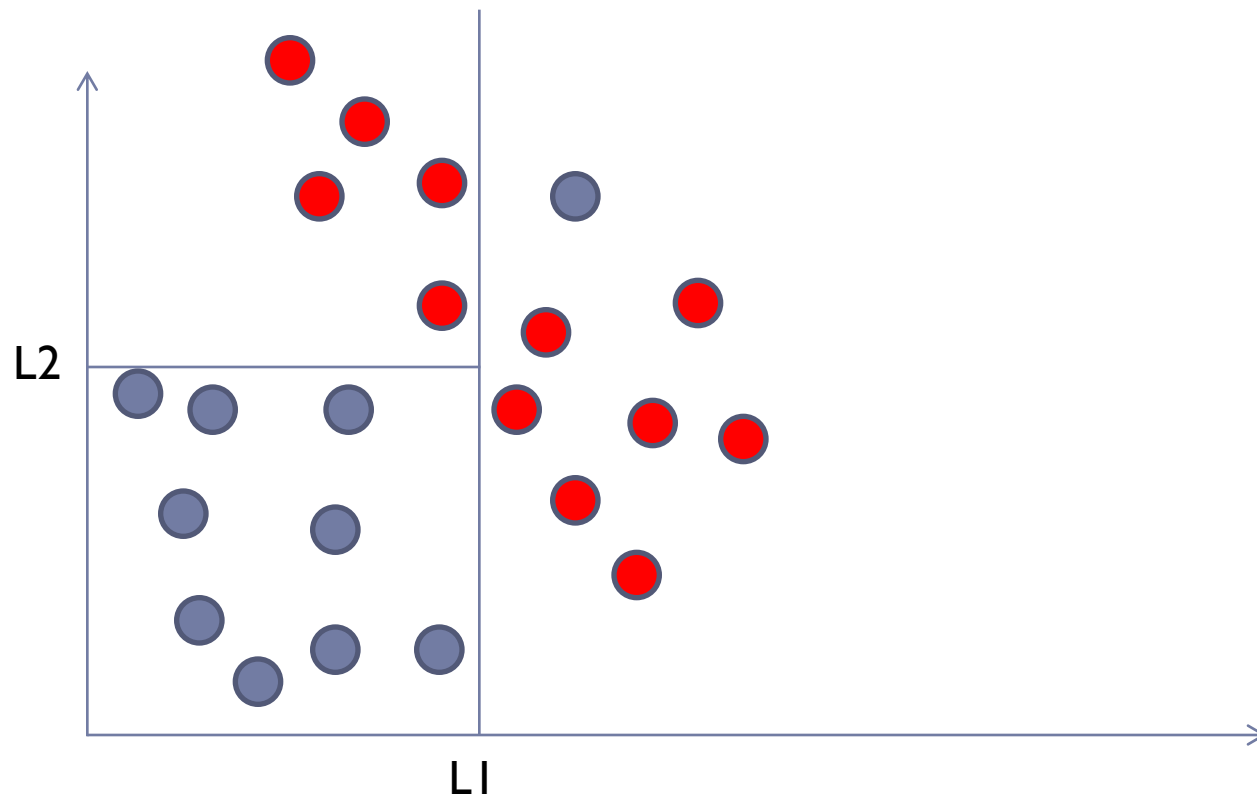
Árvore de Decisão



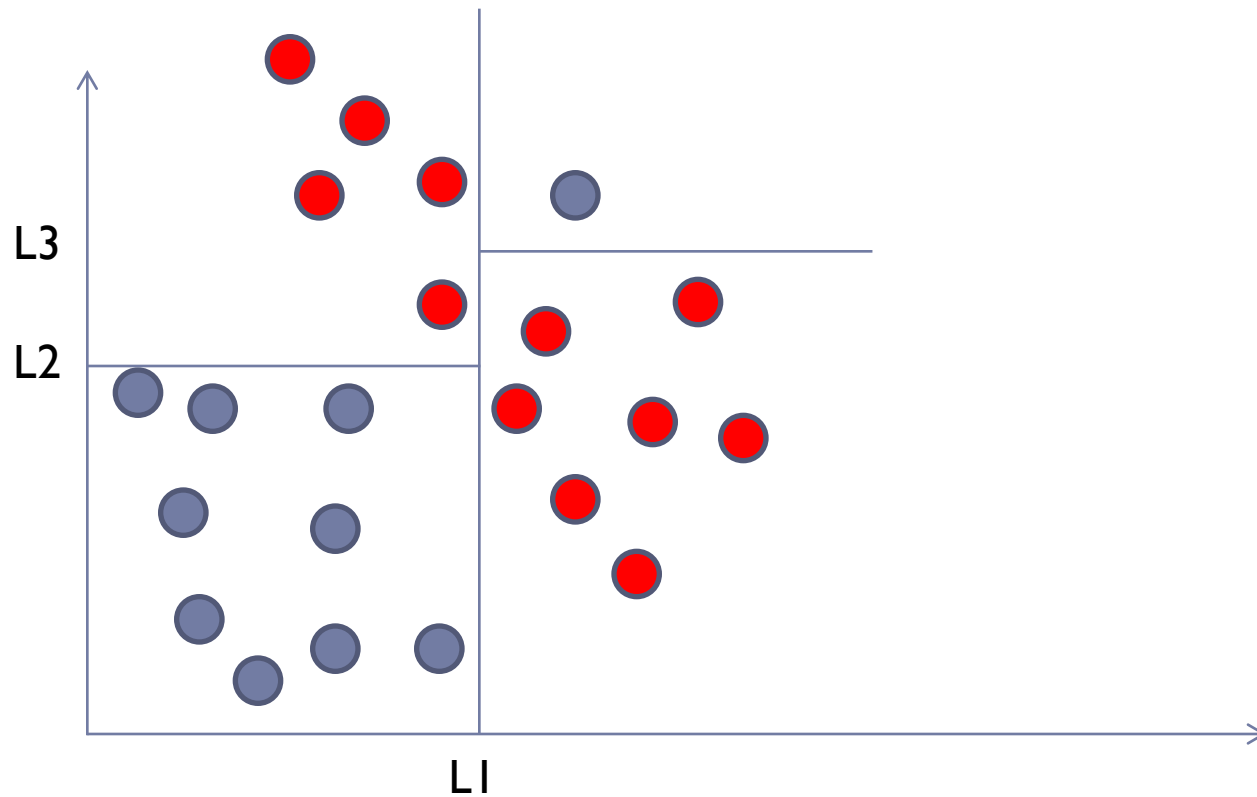
Árvore de Decisão



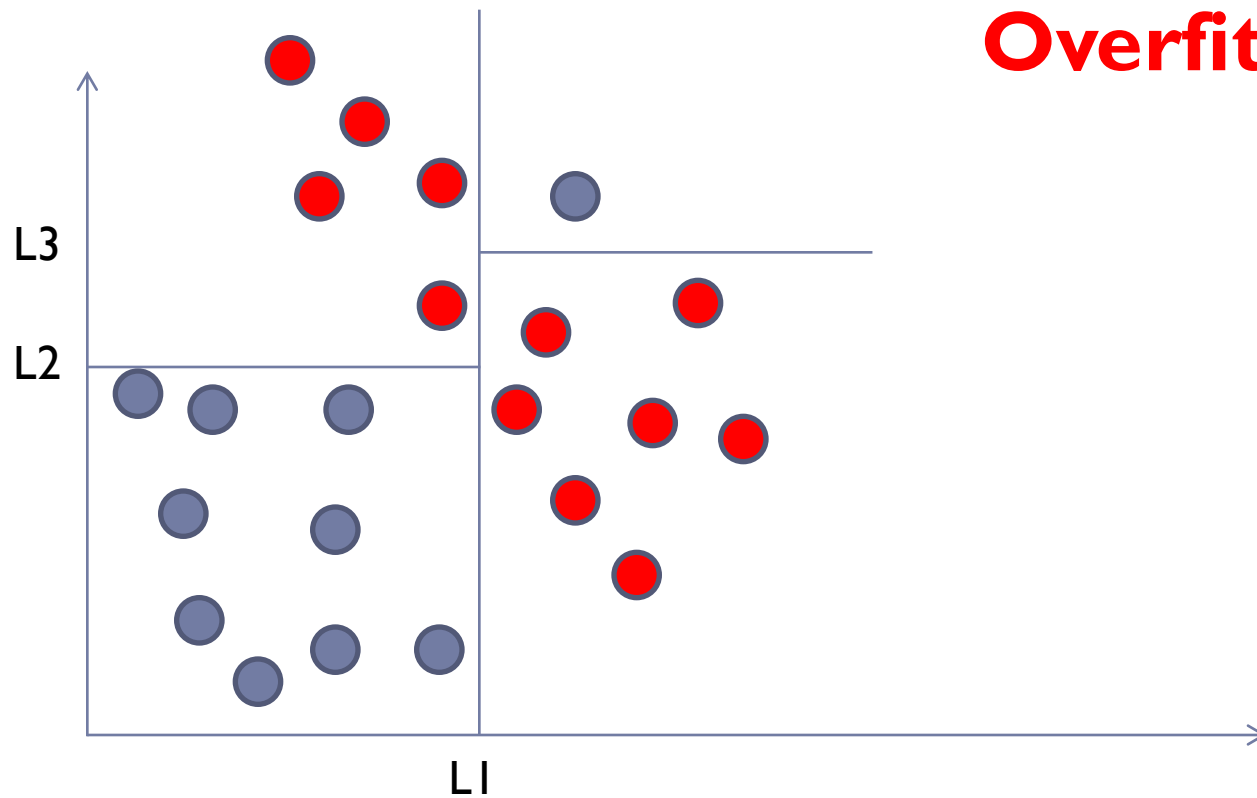
Árvore de Decisão



Árvore de Decisão



Árvore de Decisão



Overfitting



Árvore de Decisão

- ▶ Definir altura máxima
- ▶ Poda
- ▶ Parâmetros do scikit-learn
 - ▶ `max_depth` – profundidade da árvore
 - ▶ `min_samples_leaf` – número mínimo de dados em uma folha
 - ▶ `max_leaf_nodes` – número de folhas em uma árvore



Árvore de Decisão

- ▶ **Vantagens**

- ▶ Dados com diferentes tipos de atributos (reais, categóricos ...)
- ▶ Interpretabilidade

- ▶ **Desvantagens**

- ▶ Suscetíveis a overfitting
- ▶ Não consegue modelar relações complexas entre atributos



Exercício

- ▶ Carregar os dados do dataset breast cancer
- ▶ Classificar usando arvores de decisão

```
from sklearn.datasets import load_breast_cancer  
cancer = load_breast_cancer()  
(X_cancer, y_cancer) = load_breast_cancer(return_X_y = True)
```



Pré-Processamento

Pré-processamento

- ▶ Em muitas aplicações é preciso modificar os dados originais
 - ▶ Anomalias
 - ▶ Valores faltantes
 - ▶ Tipos de dados
 - ▶ Atributos categóricos



Valores Faltantes

- ▶ Resultado de diversos fatores presentes em aplicações reais
 - ▶ Falhas de coleta
 - ▶ Falha de sensor
 - ▶ Falha de gravação
- ▶ Estratégias comuns
 - ▶ Descartar dados
 - ▶ Imputar
 - ▶ Valores constantes
 - ▶ Media dos dados



Imputação

```
import numpy as np
import pandas as pd

fruits = pd.read_table('fruit_data_with_colors_miss.txt', na_values=['?', '.'])
pd.isnull(fruits)
fruits.fillna(0)
fruits['fruit_subtype'].fillna('missing')
fruits[['mass', 'width', 'height', 'color_score']].fillna(fruits.mean())
```



Atributos categóricos

- ▶ Diferentes de atributos discretos
- ▶ Não possuem ordem natural
- ▶ One-hot encoding

1	'red', 'red', 'green'
---	-----------------------

1	0, 0, 1
---	---------

1	[1, 0]
---	--------

2	[1, 0]
---	--------

3	[0, 1]
---	--------



One-hot encoding

```
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import OneHotEncoder

fruits = pd.read_table('fruit_data_with_colors.txt')

data = fruits['fruit_subtype']
label_encoder = LabelEncoder()
integer_encoded = label_encoder.fit_transform(data)
print(integer_encoded)
```

```
onehot_encoder = OneHotEncoder(sparse=False)
integer_encoded = integer_encoded.reshape(len(integer_encoded), 1)
onehot_encoded = onehot_encoder.fit_transform(integer_encoded)
print(onehot_encoded)
```

