

# Introdução ao Aprendizado de Máquina

Lucas Gonçalves de Moura Leite

# Apresentação

---

- ▶ **Lucas Gonçalves de Moura Leite**
  - ▶ Graduado em Ciência da Computação (UFC 2009-2013)
  - ▶ Mestrado em Ciência da Computação (UFC 2014-2016)
    - ▶ Lógica - Sistemas Multiagentes
  - ▶ Doutorado em andamento Ciência da Computação (UFC 2017-)
    - ▶ Aprendizado de Máquina
  - ▶ 2016 – 2018
    - ▶ Laboratório de Sistemas e Banco de Dados – Aprendizado de máquina para predição de falhas em discos rígidos
  - ▶ 2018 – Presente
    - ▶ Instituto Atlântico
    - ▶ Professor



# Aprendizado de Máquina

---

- ▶ **Aprendizado de Máquina** (*Machine Learning*)
  - ▶ “Dotar o computador da capacidade de realizar uma tarefa sem que este tenha sido explicitamente programado para isso”



# Aprendizado de Máquina

---

- ▶ **Aprendizado de Máquina** (*Machine Learning*)
  - ▶ “Dotar o computador da capacidade de realizar uma tarefa sem que este tenha sido explicitamente programado para isso”
  - ▶ Aprender através da experiência (aprender com os erros)



# Aprendizado de Máquina

---

- ▶ **Aprendizado de Máquina** (*Machine Learning*)
  - ▶ “Dotar o computador da capacidade de realizar uma tarefa sem que este tenha sido explicitamente programado para isso”
  - ▶ Aprender através da experiência (aprender com os erros)
  - ▶ Experiência = Dados (exemplos)



# Aprendizado de Máquina

---

- ▶ O estudo de programas que podem aprender a partir de exemplos
- ▶ Algoritmos de AM tem a capacidade de generalizar a partir de uma série de exemplos



# Aprendizado de Máquina

---

- ▶ Realizar tarefas através da estratégia tradicional de fazer código
  - ▶ Baseado na utilização de regras
- ▶ Tarefas difíceis
  - ▶ Reconhecimento de face
  - ▶ Reconhecimento de sentenças (*speech to text*)



# Aprender a partir de exemplos

---

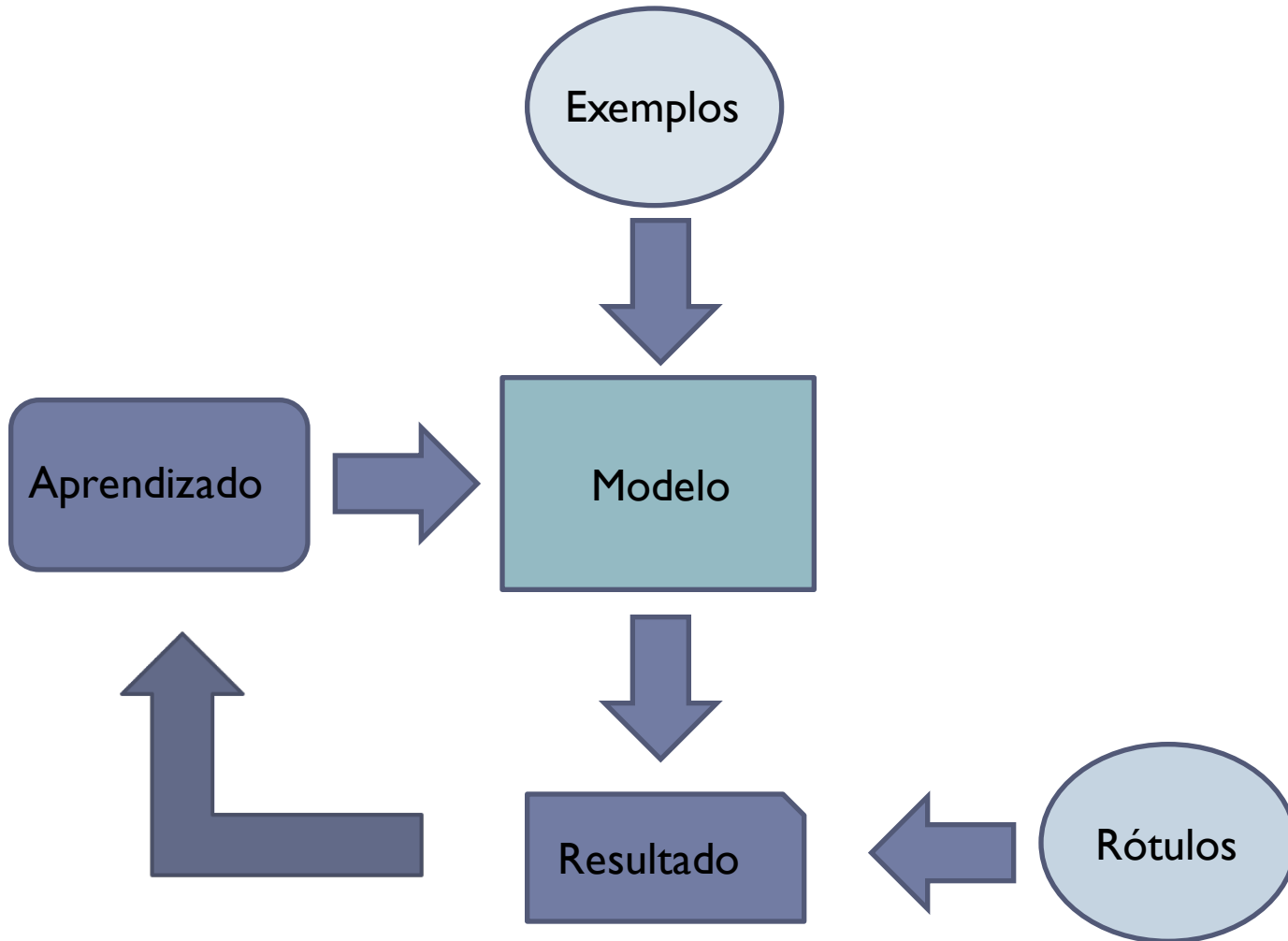
- ▶ Geração de um conjunto de exemplos rotulados
- ▶ Conjunto de treinamento
- ▶ Aprender regras e utilizar para exemplos não vistos (capacidade de generalização)





# Processo de Treinamento

---



# Aprendizado de Máquina

---

- ▶ Estatística
- ▶ Computação
- ▶ Otimização



# Neste Curso

---

- ▶ Entender os conceitos básicos de AM
- ▶ Como aplicar alguns dos algoritmos AM mais populares
- ▶ Introdução ao scikit-learn (python)
- ▶ Para maior aprofundamento nos conceitos apresentados aqui
  - ▶ Machine Learning ([coursera.org](https://www.coursera.org))



# Aprendizado de Máquina

---

- ▶ **Paradigmas de Aprendizado**
  - ▶ Aprendizado Supervisionado
  - ▶ Aprendizado Não-Supervisionado



# Aprendizado Supervisionado

---

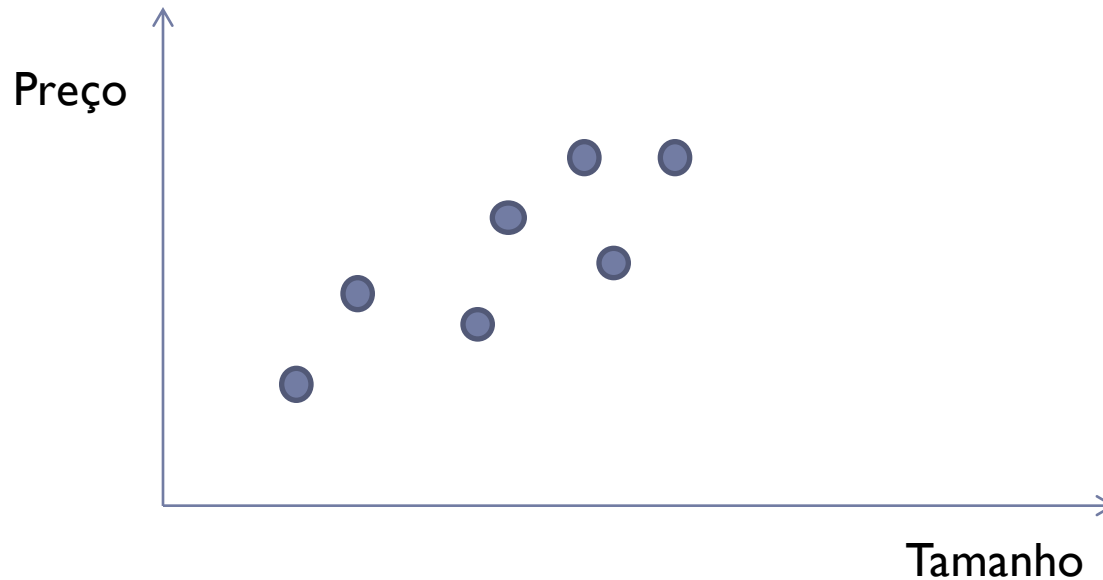
- ▶ Aprender através de dados rotulados



# Aprendizado Supervisionado

---

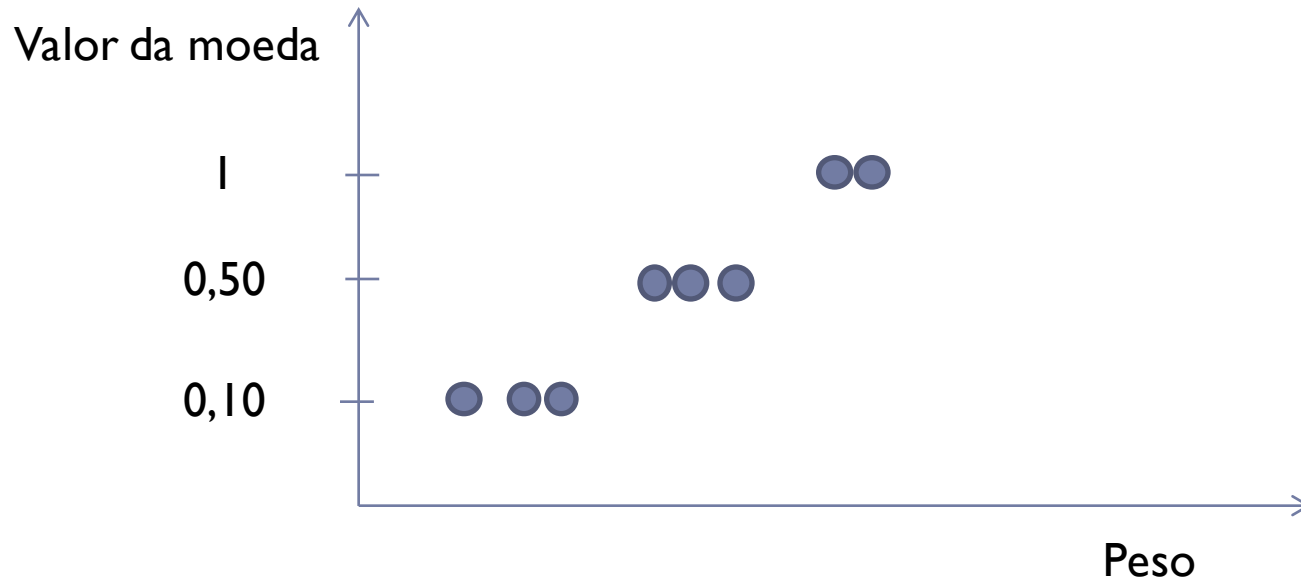
- ▶ Aprender através de dados rotulados
  - ▶ Exemplo
    - ▶ Preço de imóveis



# Aprendizado Supervisionado

---

- ▶ Aprender através de dados rotulados
  - ▶ Exemplo
    - ▶ Máquina de refrigerante



# Aprendizado Não-Supervisionado

---

- ▶ Aprender através de dados não rotulados

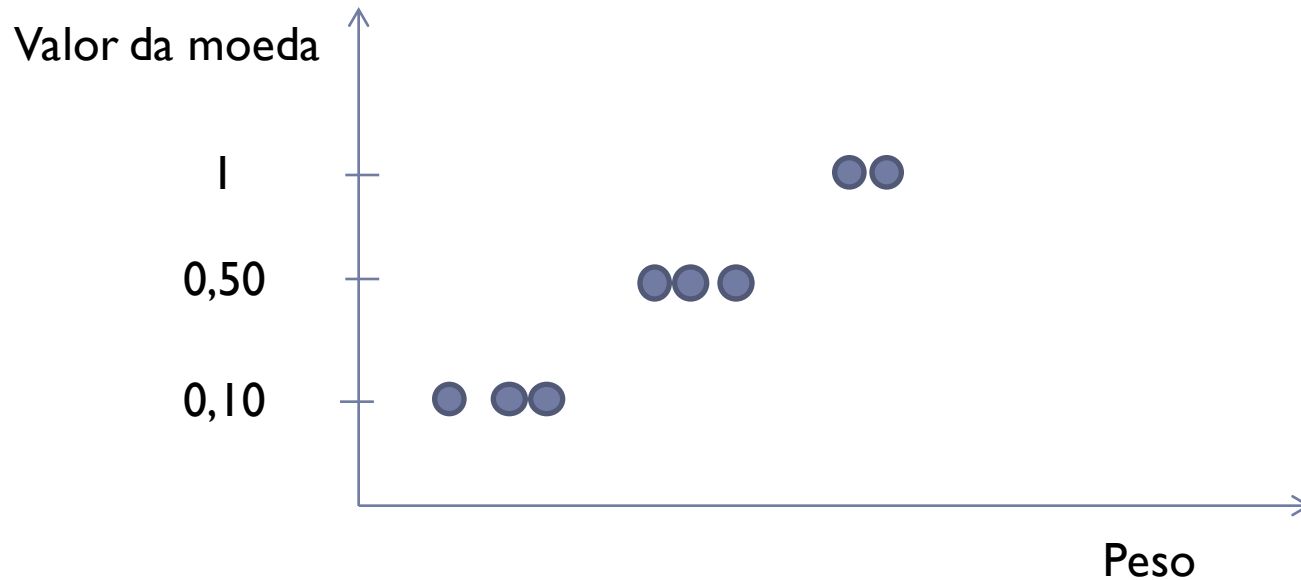




# Aprendizado Não-Supervisionado

---

- ▶ Aprender através de dados não rotulados
  - ▶ Exemplo



# Aprendizado Não-Supervisionado

---

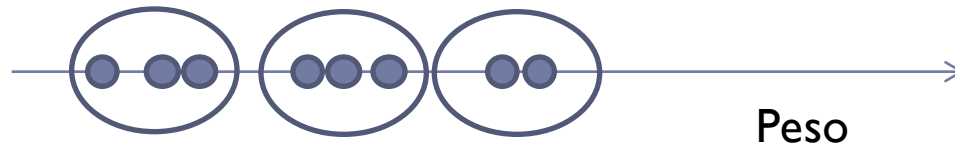
- ▶ Aprender através de dados não rotulados
  - ▶ Exemplo



# Aprendizado Não-Supervisionado

---

- ▶ Aprender através de dados não rotulados
  - ▶ Exemplo



# Como aplicar AM

---

- ▶ **Representação**

- ▶ Escolha dos atributos
- ▶ Escolha do classificador

- ▶ **Avaliação**

- ▶ Critério de avaliação do seu método

- ▶ **Otimização**

- ▶ Encontrar os melhores parâmetros para otimizar o critério de desempenho



# Representação

## ► Extração de atributos (*features*)

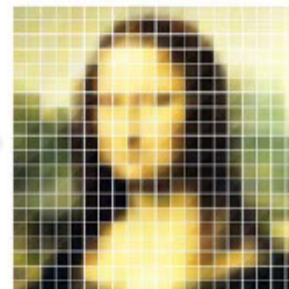
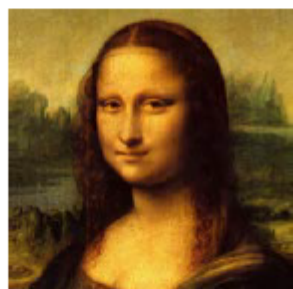
Email

To: Chris Brooks  
From: Daniel Romero  
Subject: Next course offering  
Hi Daniel,  
Could you please send the outline for the  
next course offering? Thanks! -- Chris



<u>Feature</u>	<u>Count</u>
to	1
chris	2
brooks	1
from	1
daniel	2
romero	1
the	2
...	

Imagem



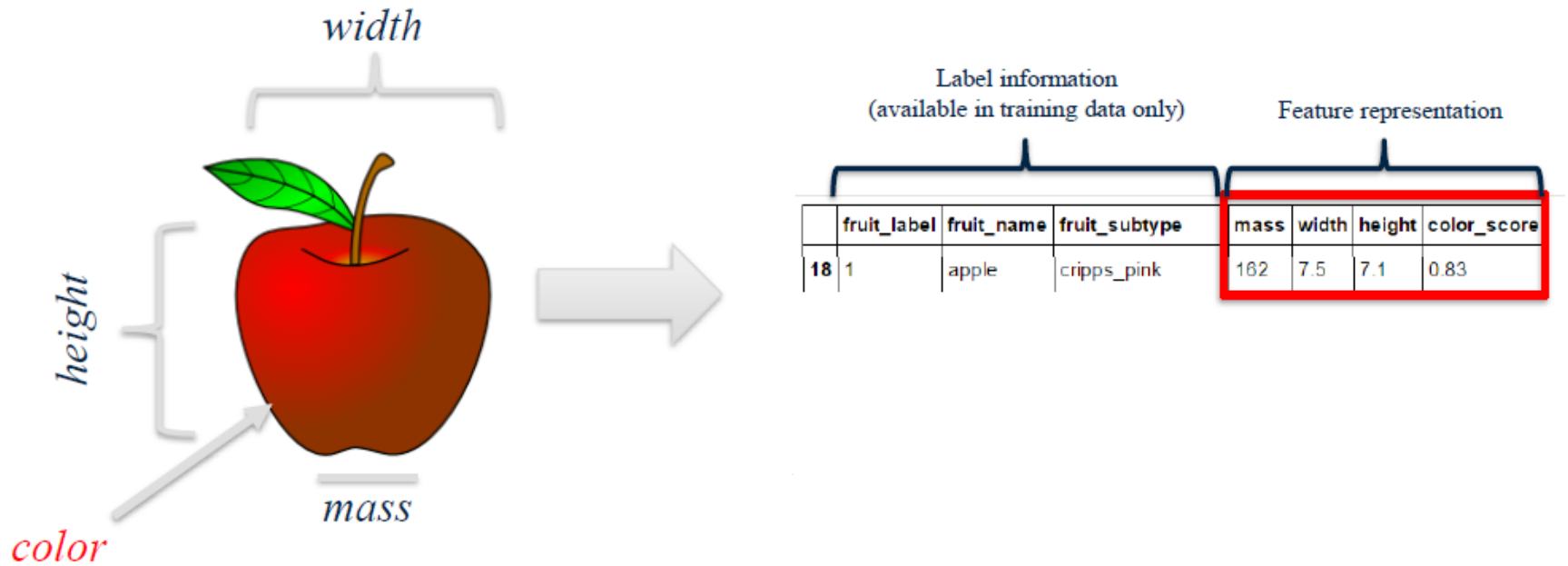
Peixe



<u>Feature</u>	<u>Value</u>
DorsalFin	Yes
MainColor	Orange
Stripes	Yes
StripeColor1	White
StripeColor2	Black
Length	4.3 cm

# Atributos

---



# Representação

---

- ▶ Escolha do método de AM
  - ▶ Muitas opções
  - ▶ Critérios
    - ▶ Complexidade
    - ▶ Interpretabilidade
    - ▶ Custo computacional



# Avaliação

---

- ▶ Como verificar a qualidade do seu modelo
  - ▶ Vários Critérios
    - ▶ Natureza do tarefa (classificação, regressão ...)
    - ▶ Características dos dados
    - ▶ Natureza da aplicação





# Otimização

---

- ▶ Performance do modelo depende das escolhas da etapa de representação
  - ▶ Atributos
    - ▶ Quantos e quais
  - ▶ Método de AM
    - ▶ Natureza e parâmetros



# AM com Python

---

- ▶ **Distribuição Anaconda**
  - ▶ Scikit Learn
  - ▶ Pandas
  - ▶ Numpy
  - ▶ SciPy
  - ▶ Matplotlib



# Scikit Learn

---

## scikit-learn: Python Machine Learning Library



- scikit-learn Homepage  
<http://scikit-learn.org/>
- scikit-learn User Guide  
[http://scikit-learn.org/stable/user\\_guide.html](http://scikit-learn.org/stable/user_guide.html)
- scikit-learn API reference  
<http://scikit-learn.org/stable/modules/classes.html>
- In Python, we typically import classes and functions we need like this:

```
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
```



## SciPy Library: Scientific Computing Tools



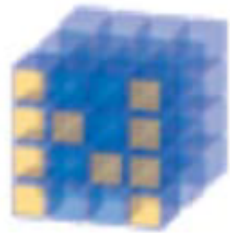
<http://www.scipy.org/>

- Provides a variety of useful scientific computing tools, including statistical distributions, optimization of functions, linear algebra, and a variety of specialized mathematical functions.
- With scikit-learn, it provides support for *sparse matrices*, a way to store large tables that consist mostly of zeros.
- Example import: `import scipy as sp`

# Numpy

---

## NumPy: Scientific Computing Library



<http://www.numpy.org/>

- Provides fundamental data structures used by scikit-learn, particularly multi-dimensional arrays.
- Typically, data that is input to scikit-learn will be in the form of a NumPy array.
- Example import: `import numpy as np`



# Pandas

---

## Pandas: Data Manipulation and




- Provides key data structures like `DataFrame`
- Also, support for reading/writing data in different formats
- Example import: `import pandas as pd`



# Matplotlib

---

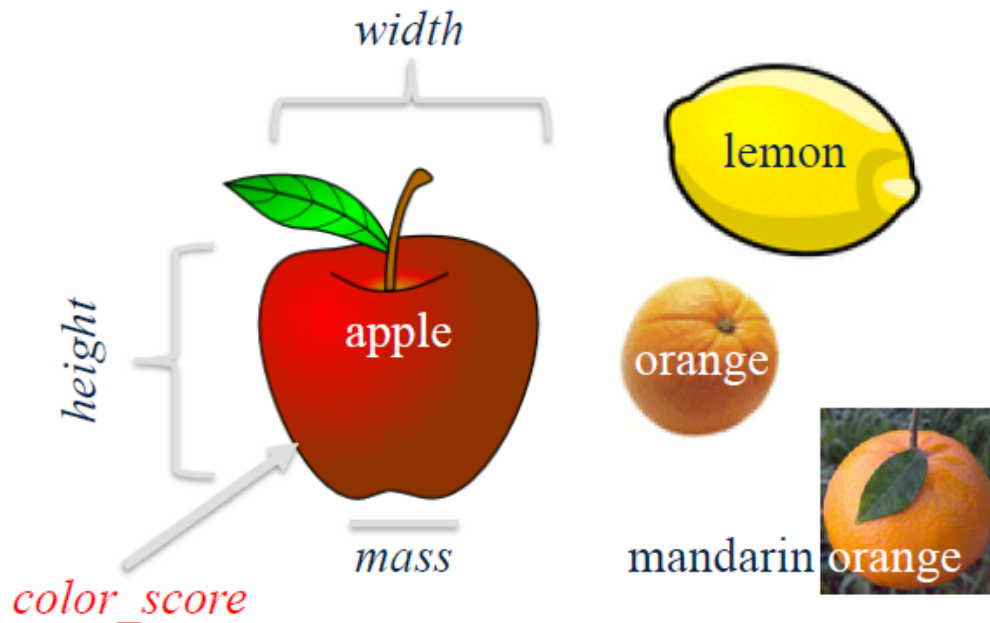
## matplotlib and other plotting libraries

**matplotlib**  <http://matplotlib.org/>

- We typically use matplotlib's **pyplot** module:  
`import matplotlib.pyplot as plt`
- We also sometimes use the **seaborn** visualization library (<http://seaborn.pydata.org/>)  
`import seaborn as sn`
- And sometimes the **graphviz** plotting library:  
`import graphviz`



# Uma Aplicação Simples



	fruit_label	fruit_name	fruit_subtype	mass	width	height	color_score
0	1	apple	granny_smith	192	8.4	7.3	0.55
1	1	apple	granny_smith	180	8.0	6.8	0.59
2	1	apple	granny_smith	176	7.4	7.2	0.60
3	2	mandarin	mandarin	86	6.2	4.7	0.80
4	2	mandarin	mandarin	84	6.0	4.6	0.79
5	2	mandarin	mandarin	80	5.8	4.3	0.77
6	2	mandarin	mandarin	80	5.9	4.3	0.81
7	2	mandarin	mandarin	76	5.8	4.0	0.81
8	1	apple	brae_bum	178	7.1	7.8	0.92
9	1	apple	brae_bum	172	7.4	7.0	0.89
10	1	apple	brae_bum	166	6.9	7.3	0.93
11	1	apple	brae_bum	172	7.1	7.6	0.92
12	1	apple	brae_bum	154	7.0	7.1	0.88
13	1	apple	golden_delicious	164	7.3	7.7	0.70
14	1	apple	golden_delicious	152	7.6	7.3	0.69
15	1	apple	golden_delicious	156	7.7	7.1	0.69
16	1	apple	golden_delicious	156	7.6	7.5	0.67

fruit\_data\_with\_colors.txt



# Dados

Rótulo ou Label

Atributos ou  
features

Exemplo ou  
instância

	fruit_label	fruit_name	fruit_subtype	mass	width	height	color_score
0	1	apple	granny_smith	192	8.4	7.3	0.55
1	1	apple	granny_smith	180	8.0	6.9	0.59
2	1	apple	granny_smith	176	7.4	7.2	0.60
3	2	mandarin	mandarin	86	6.2	4.7	0.80
4	2	mandarin	mandarin	84	6.0	4.6	0.79
5	2	mandarin	mandarin	80	5.8	4.3	0.77
6	2	mandarin	mandarin	80	5.9	4.3	0.81
7	2	mandarin	mandarin	76	5.8	4.0	0.81
8	1	apple	braeburn	178	7.1	7.8	0.92
9	1	apple	braeburn	172	7.4	7.0	0.89
10	1	apple	braeburn	166	6.9	7.3	0.93
11	1	apple	braeburn	172	7.1	7.6	0.92
12	1	apple	braeburn	154	7.0	7.1	0.88
13	1	apple	golden_delicious	164	7.3	7.7	0.70
14	1	apple	golden_delicious	152	7.6	7.3	0.69
15	1	apple	golden_delicious	156	7.7	7.1	0.69
16	1	apple	golden_delicious	156	7.6	7.5	0.67
17	1	apple	golden_delicious	168	7.5	7.6	0.73
18	1	apple	cripps_pink	162	7.5	7.1	0.83
19	1	apple	cripps_pink	162	7.4	7.2	0.85
20	1	apple	cripps_pink	160	7.5	7.5	0.86

# Carregando os dados

---

```
%matplotlib notebook
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.model_selection import train_test_split

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

```
fruits.shape
```

```
(59, 7)
```

```
fruits.head(2)
```

	fruit_label	fruit_name	fruit_subtype	mass	width	height	color_score
0	1	apple	granny_smith	192	8.4	7.3	0.55
1	1	apple	granny_smith	180	8.0	6.8	0.59



# Estruturas de Dados

---

- ▶ **Python**

- ▶ Listas (Vetores e Matrizes)
  - ▶ Dicionário

- ▶ **Pandas**

- ▶ **Series**

- ▶ Lista e dicionário
    - ▶ Itens ordenados e existem labels que podem ser usados para fazer consultas

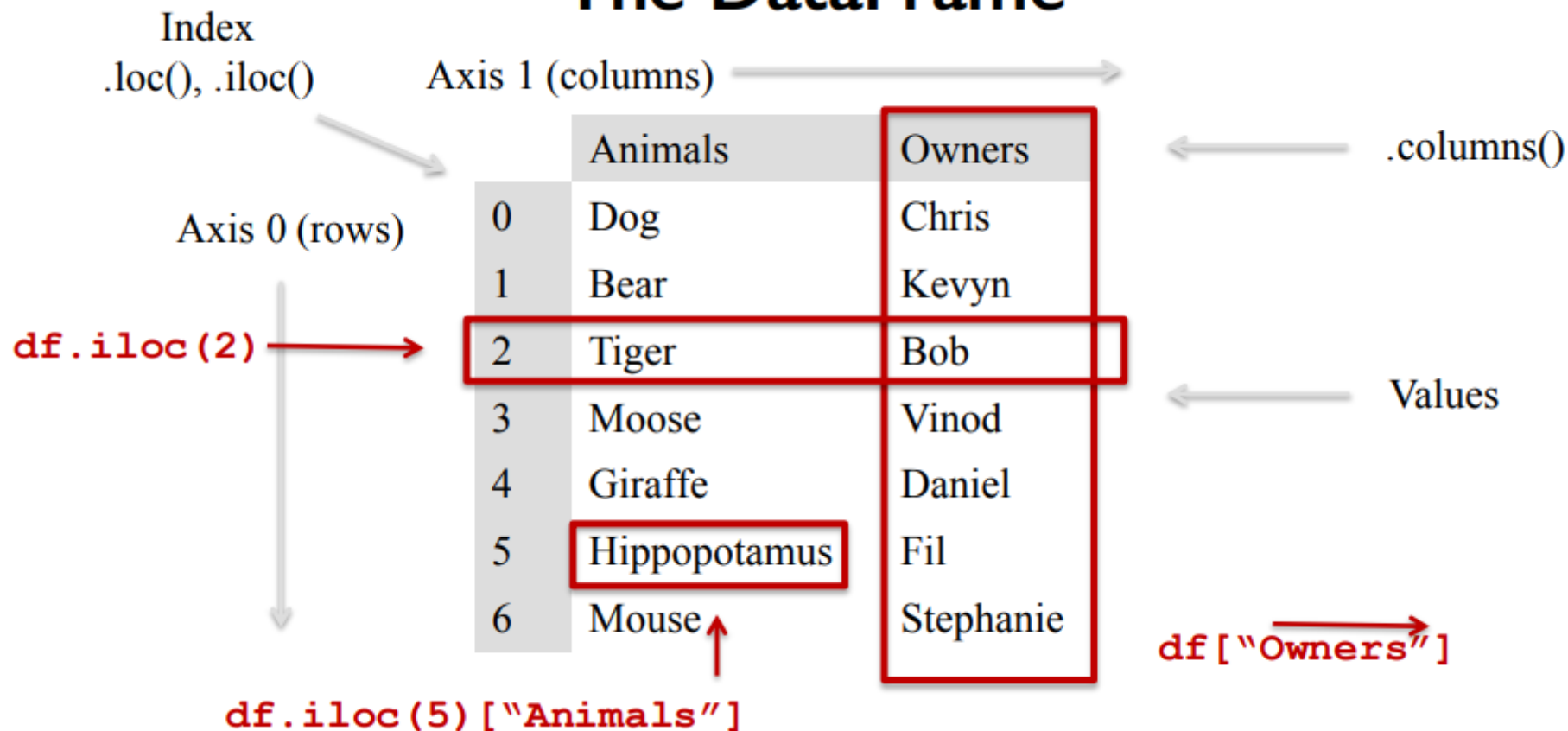
- ▶ **DataFrame (.DataFrame)**

- ▶ Multidimensional
      - ▶ Itens ordenados
      - ▶ Possibilita consulta em linhas e colunas
- 



# DataFrame

## The DataFrame



# Selecionando linhas e colunas

	fruit_label	fruit_name	fruit_subtype	mass	width	height	color_score
0	1	apple	granny_smith	192	8.4	7.3	0.55
1	1	apple	granny_smith	180	8.0	6.8	0.59
2	1	apple	granny_smith	176	7.4	7.2	0.60
3	2	mandarin	mandarin	86	6.2	4.7	0.80
4	2	mandarin	mandarin	84	6.0	4.6	0.79

```
fruits['mass']  
fruits[:3]  
fruits['mass'][:3]  
freq = fruits['fruit_name'].value_counts()  
freq.plot(kind = 'bar')  
plt.show()
```

Coluna 'mass'

3 primeiras linhas

3 primeiras linhas da coluna  
'mass'

Contagem de ocorrências do  
atributo 'fruit\_name'

# Selecionando linhas e colunas

---

```
macas = fruits['fruit_name'] == 'apple'  
fruits[fruits['fruit_name'] == 'apple']  
pesadas = fruits['mass'] > 175  
fruits[macas & pesadas]
```

Indices das maçãs

Instancias cujo atributo  
'fruit\_name' é maçã

Indices das frutas que pesam  
mais que 175

Maçãs que pesam mais que 175



# Plotando os dados

---

```
X1 = fruits['width']  
X2 = fruits['height']  
plt.scatter(X1, X2, c = y)  
plt.show()  
pd.scatter_matrix(X, c= y, figsize=(9,9))
```



# Treino x Teste

---

- ▶ **Aprendizado de máquina**
  - ▶ Capacidade de generalização
  - ▶ Como verificar ?





# Treino x Teste

---

```
# plotting a scatter matrix  
from matplotlib import cm  
  
X = fruits[['height', 'width', 'mass', 'color_score']]  
y = fruits['fruit_label']  
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=0)
```



# Treino x Teste

X					Y		X_train					y_train		X_test					y_test	
height	width	mass	color_score				height	width	mass	color_score				height	width	mass	color_score			
0	7.3	8.4	192	0.55	0	1	42	7.2	7.2	154	0.82	42	3	26	9.2	9.6	362	0.74	26	3
1	6.8	8.0	180	0.59	1	1	48	10.1	7.3	174	0.72	48	4	35	7.9	7.1	150	0.75	35	3
2	7.2	7.4	176	0.60	2	1	7	4.0	5.8	76	0.81	7	2	43	10.3	7.2	194	0.70	43	4
3	4.7	6.2	86	0.80	3	2	14	7.3	7.6	152	0.69	14	1	28	7.1	6.7	140	0.72	28	3
4	4.6	6.0	84	0.79	4	2	32	7.0	7.2	164	0.80	32	3	11	7.6	7.1	172	0.92	11	1
5	4.3	5.8	80	0.77	5	2	49	8.7	5.8	132	0.73	49	4	2	7.2	7.4	176	0.60	2	1
6	4.3	5.9	80	0.81	6	2	29	7.4	7.0	160	0.81	29	3	34	7.8	7.6	142	0.75	34	3
7	4.0	5.8	76	0.81	7	2	37	7.3	7.3	151	0.79	37	3	46	10.2	7.3	216	0.71	46	4
8	7.8	7.1	178	0.92	8	1	56	7.3	7.3	151	0.73	56	4	40	7.5	7.1	154	0.78	40	3
9	7.0						39	7.4	6.8	144	0.75	39	3	22	7.1	7.3	140	0.87	22	1
10	7.3						3	4.7	6.2	86	0.80	3	2	4	4.6	6.0	84	0.79	4	2
11	7.6						0	7.3	8.4	192	0.55	0	1	30	7.9	7.1	150	0.75	30	3
12	7.1						53	8.4	6.0	120	0.74	53	4	41	8.2	7.6	180	0.79	41	3
13	7.7						47	9.7	7.3	196	0.72	47	4	33	8.1	7.5	190	0.74	33	3
14	7.3	7.6	152	0.69	14	1	44	10.5	7.3	200	0.72	44	4							
15	7.1	7.7	156	0.69	15	1														
16	7.5	7.6	156	0.67	16	1														
17	7.6	7.5	168	0.73	17	1														
18	7.1	7.5	162	0.83	18	1														
19	7.2	7.4	162	0.85	19	1														

`X_train, X_test, y_train, y_test  
= train_test_split(X, y)`

Dado Original

Conjunto de  
Treino

Conjunto de  
Teste

# Primeiro Classificador (K-NN)

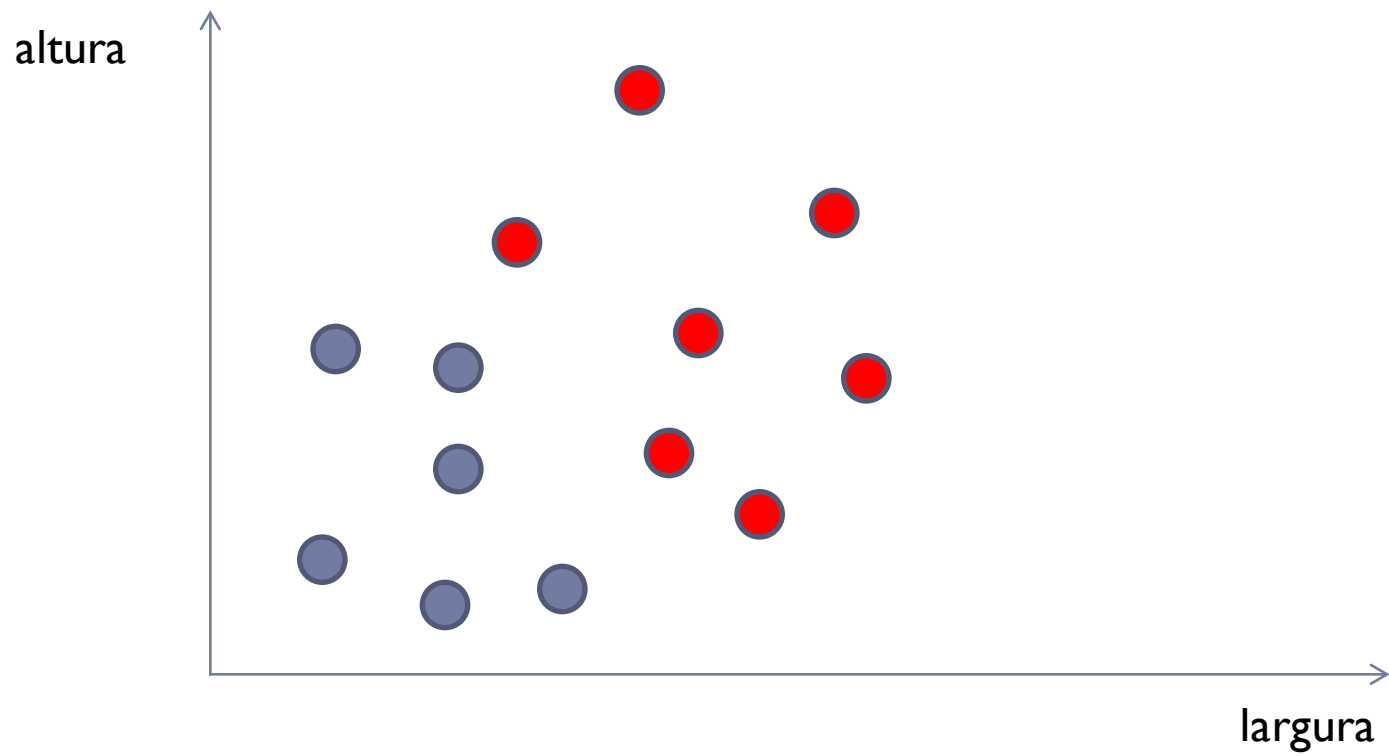
---

- ▶ Para um novo dado, verifica-se quais dados do conjunto de treinamento são mais semelhantes.
  - ▶ Atributos ( $x$ )
- ▶ A classe escolhida é a classe da maioria dos dados mais próximos



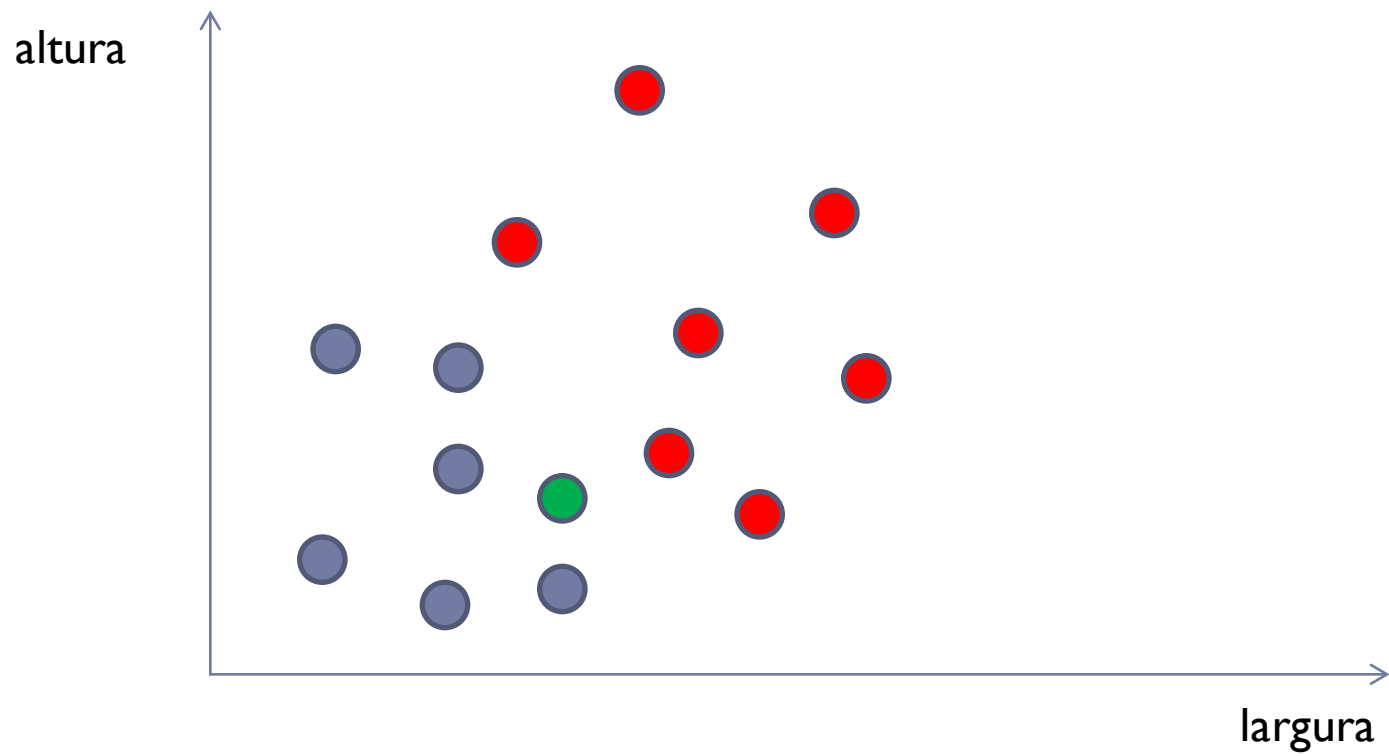
# K-NN

---



# K-NN

---



# K-NN - Classificação

---

- ▶ Armazena  $n$  exemplos de cada classe (conjunto de treinamento)
- ▶ Calcula a distância do novo ponto para todos os  $n$  exemplos de cada classe



# K-NN - Classificação

---

- ▶ Armazena  $n$  exemplos de cada classe (conjunto de treinamento)
- ▶ Calcula a distância do novo ponto para todos os  $n$  exemplos de cada classe
- ▶ Verifica quais são os  $k$  pontos mais próximos



# K-NN - Classificação

---

- ▶ Armazena  $n$  exemplos de cada classe (conjunto de treinamento)
- ▶ Calcula a distância do novo ponto para todos os  $n$  exemplos de cada classe
- ▶ Verifica quais são os  $k$  pontos mais próximos
- ▶ Verifica a classe destes  $k$  pontos





# K-NN - Classificação

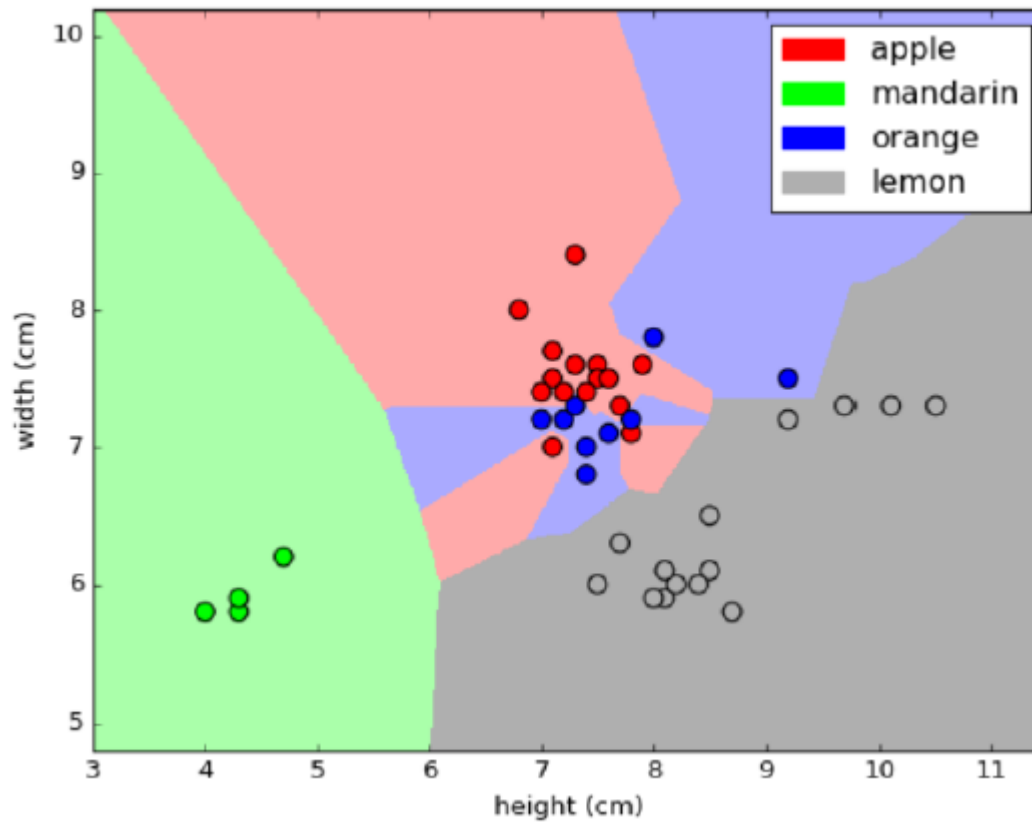
---

- ▶ Armazena  $n$  exemplos de cada classe (conjunto de treinamento)
- ▶ Calcula a distância do novo ponto para todos os  $n$  exemplos de cada classe
- ▶ Verifica quais são os  $k$  pontos mais próximos
- ▶ Verifica a classe destes  $k$  pontos
- ▶ A classe do novo dado será igual a classe da maioria



# K-NN para as frutas (K=1)

---



# K-NN no Scikit Learn

---

```
from sklearn.neighbors import KNeighborsClassifier
```

```
knn = KNeighborsClassifier(n_neighbors = 5)
```

```
knn.fit(X_train, y_train)
```

```
knn.score(X_test, y_test)
```

```
# first example: a small fruit with mass 20g, width 4.3 cm, height 5.5 cm  
fruit_prediction = knn.predict([[20, 4.3, 5.5]])  
print(fruit_prediction)
```

```
# first example: a small fruit with mass 20g, width 4.3 cm, height 5.5 cm  
fruit_prediction = knn.predict([[20, 4.3, 5.5]])  
print(fruit_prediction)
```

```
fruit_prediction = knn.predict(X_test.iloc[0])  
print(fruit_prediction)
```

```
y_test.iloc[0]  
#lookup_fruit_name[fruit_prediction[0]]
```



# Superfície de Classificação

---

```
from adspy_shared_utilities import plot_fruit_knn  
plot_fruit_knn(X_train, y_train, 5, 'uniform')  # we choose 5 nearest neighbors
```

- ▶ Variem o valor de k



# Efeito de K na acurácia

---

```
k_range = range(1,20)
scores = []

for k in k_range:
    knn = KNeighborsClassifier(n_neighbors = k)
    knn.fit(X_train, y_train)
    scores.append(knn.score(X_test, y_test))

plt.figure()
plt.xlabel('k')
plt.ylabel('accuracy')
plt.scatter(k_range, scores)
plt.xticks([0,5,10,15,20]);
```



# Exercício

---

- ▶ Carregar os dados do dataset iris
- ▶ Visualizar os dados (escolher dois atributos)
- ▶ Fazer o scatterplot
- ▶ Classificar usando K-NN (variar distancia e k)
- ▶ Visualizar as superfícies de decisão

```
import numpy as np
import pandas as pd
from sklearn.datasets import load_iris
```



# Exercício

---

- ▶ Carregar os dados do dataset iris
- ▶ Visualizar os dados (escolher dois atributos)
- ▶ Fazer o scatterplot
- ▶ Classificar usando K-NN (variar distancia e k)
- ▶ Visualizar as superfícies de decisão

```
import numpy as np
import pandas as pd
from sklearn.datasets import load_iris

data1 = pd.DataFrame(data= np.c_[iris['data'], iris['target']],
                     columns= iris['feature_names'] + ['target'])
```



# K-NN

---

- ▶ **Vantagens**

- ▶ Simples
- ▶ Treinamento rápido

- ▶ **Desvantagens**

- ▶ Custo computacional do teste
- ▶ Influenciado pela ordem de grandeza dos atributos

