


Inteligencia Artificial con Python y scikit-learn

[Install](#) [User Guide](#) [API](#) [Examples](#) [Community](#) [More](#)

scikit-learn

Machine Learning in Python

[Getting Started](#)

[Release Highlights for 1.6](#)

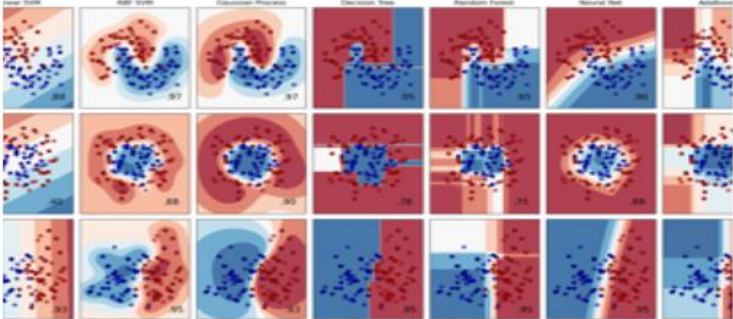
- Simple and efficient tools for predictive data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable - BSD license

Classification

Identifying which category an object belongs to.

Applications: Spam detection, image recognition.

Algorithms: [Gradient boosting](#), [nearest neighbors](#), [random forest](#), [logistic regression](#), and [more...](#)

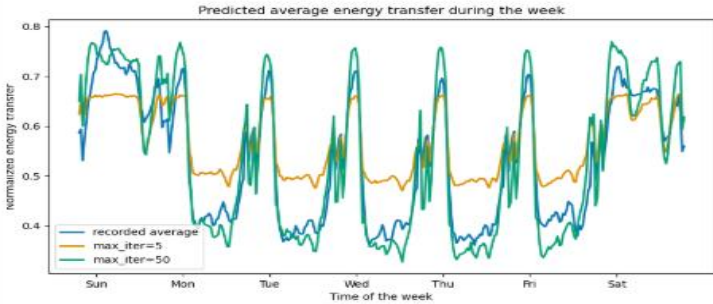


Regression

Predicting a continuous-valued attribute associated with an object.

Applications: Drug response, stock prices.

Algorithms: [Gradient boosting](#), [nearest neighbors](#), [random forest](#), [ridge](#), and [more...](#)




Clustering

Automatic grouping of similar objects into sets.

Applications: Customer segmentation, grouping experiment outcomes.

Algorithms: [k-Means](#), [HDBSCAN](#), [hierarchical clustering](#), and [more...](#)





Aprendizaje automático aplicado

K-Nearest Neighbors: Classification and Regression

K-Vecinos más cercanos: Clasificación y regresión



K-Nearest Neighbors (KNN)

- El **K-Nearest Neighbors (KNN)** es un algoritmo de aprendizaje supervisado **basado en la similitud de datos**.
- Se utiliza tanto para tareas de clasificación como de regresión, y su principio fundamental radica en identificar los **k vecinos más cercanos** de una instancia para tomar una decisión.
- Es un método **lazy learning** que no aprende un modelo explícito, sino que **almacena los datos para consultarlos directamente**.



K-Nearest Neighbors (KNN)

Predicción basada en vecinos:

- Para predecir el valor de una nueva instancia, KNN identifica las **k** instancias más cercanas en el conjunto de entrenamiento y utiliza sus valores o etiquetas para determinar la salida.
- En **clasificación**, asigna la **clase más frecuente** entre los **k** vecinos.
- En **regresión**, calcula el **promedio de los valores** de los **k** vecinos.



K-Nearest Neighbors (KNN)

Elección de k :

- El número de vecinos k es un hiperparámetro clave que afecta el rendimiento del modelo.
- **Valores pequeños de k** pueden hacer que el modelo sea más sensible al ruido (sobreajuste).
- **Valores grandes de k** pueden diluir las características locales y hacer que el modelo pierda precisión (subajuste).

K-Nearest Neighbors (KNN)

Similitud (distancia):

- La proximidad entre las instancias se mide mediante una métrica de distancia.
- Las más comunes son:

- Distancia Euclidiana:

$$d(x, y) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$$

- Distancia Manhattan:

$$d(x, y) = \sum_{i=1}^n |x_i - y_i|$$

- Distancia de Minkowski (generalización de las anteriores):

$$d(x, y) = \left(\sum_{i=1}^n |x_i - y_i|^p \right)^{1/p}$$

Algoritmo clasificador del k-vecinos más cercano (k-NN)

Dado un conjunto de entrenamiento X_{train} con etiquetas y_{train} y dada una nueva instancia x_{test} se va a clasificar:

1. Encuentre las instancias más similares (llamémoslas X_{NN}) a x_{test} que están en X_{train} .
2. Obtenga las etiquetas y_{NN} para las instancias de X_{NN}
3. Prediga la etiqueta para x_{test} combinando las etiquetas y_{NN} por ejemplo, mayoría simple de votos
 - *Normalización de características: Es crucial normalizar o estandarizar las variables para garantizar que tengan igual impacto en las métricas de distancia.*



Un algoritmo de K-vecinos más cercanos necesita cuatro cosas especificadas

1. Una métrica de distancia
2. ¿Cuántos vecinos "más cercanos" hay que mirar?
3. Función de ponderación ***opcional*** en los puntos vecinos
4. Método para agregar las clases de puntos vecinos



Un algoritmo de K-vecinos más cercanos necesita cuatro cosas especificadas

1. Una métrica de distancia

- Típicamente euclidiano (Minkowski con $p = 2$)

2. ¿Cuántos vecinos "más cercanos" hay que mirar?

- Por ejemplo, cinco

3. Función de ponderación opcional en los puntos vecinos

- Ignorar

4. Cómo agregar las clases de puntos de vecinos: **Voto por mayoría simple**

- (Clase con más representantes entre los vecinos más cercanos)



Aplicaciones

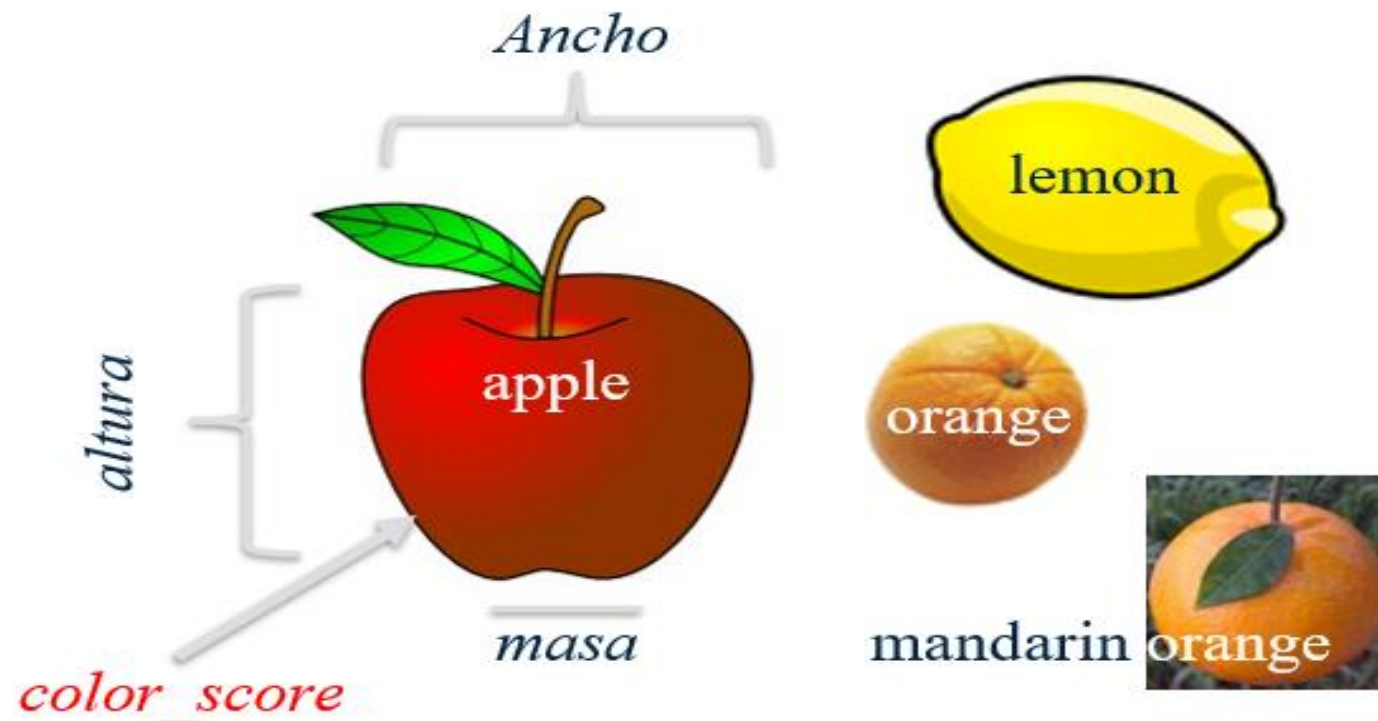
1. Clasificación:

- Reconocimiento facial.
- Clasificación de texto y detección de spam.
- Diagnóstico médico.

2. Regresión:

- Predicción de precios (inmuebles, productos).
- Modelado de tendencias de mercado.

The Fruit Dataset



	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	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

fruit_data_with_colors.txt

Créditos: Versión original del conjunto de datos de frutas creado por el Dr. Iain Murray, Univ. de Edimburgo

In [27]:

fruits

Out[27]:

	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	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

Train Test Split (Scikit-Learn + Python)



In [96]: X_train

Out[96]:

	height	width	mass	color_score
42	7.2	7.2	154	0.82
48	10.1	7.3	174	0.72
7	4.0	5.8	76	0.81
14	7.3	7.6	152	0.69
32	7.0	7.2	164	0.80
49	8.7	5.8	132	0.73
29	7.4	7.0	160	0.81
37	7.3	7.3	154	0.79
56	8.1	5.9	116	0.73
18	7.1	7.5	162	0.83
55	7.7	6.3	116	0.72
27	9.2	7.5	204	0.77
15	7.1	7.7	156	0.69
5	4.3	5.8	80	0.77
31	8.0	7.8	210	0.82
16	7.5	7.6	156	0.67

In [98]: y_train

Out[98]:

```
42 3
48 4
7 2
14 1
32 3
49 4
29 3
37 3
56 4
18 1
55 4
27 3
15 1
5 2
31 3
16 1
50 4
20 1
51 4
8 1
13 1
25 3
17 1
58 4
57 4
52 4
38 3
1 1
12 1
45 4
24 3
6 2
```

In [97]: X_test

Out[97]:

	height	width	mass	color_score
26	9.2	9.6	362	0.74
35	7.9	7.1	150	0.75
43	10.3	7.2	194	0.70
28	7.1	6.7	140	0.72
11	7.6	7.1	172	0.92
2	7.2	7.4	176	0.60
34	7.8	7.6	142	0.75
46	10.2	7.3	216	0.71
40	7.5	7.1	154	0.78
22	7.1	7.3	140	0.87
4	4.6	6.0	84	0.79
10	7.3	6.9	166	0.93
30	7.5	7.1	158	0.79
41	8.2	7.6	180	0.79
33	8.1	7.5	190	0.74

In [99]: y_test

Out[99]:

```
26 3
35 3
43 4
28 3
11 1
2 1
34 3
46 4
40 3
22 1
4 2
10 1
30 3
41 3
33 3
```


Un diagrama de dispersión de características por pares visualiza los datos utilizando todos los pares posibles de características, con un diagrama de dispersión por par de características e histogramas para cada característica

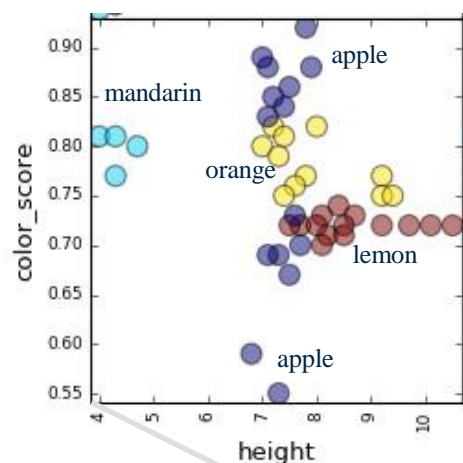
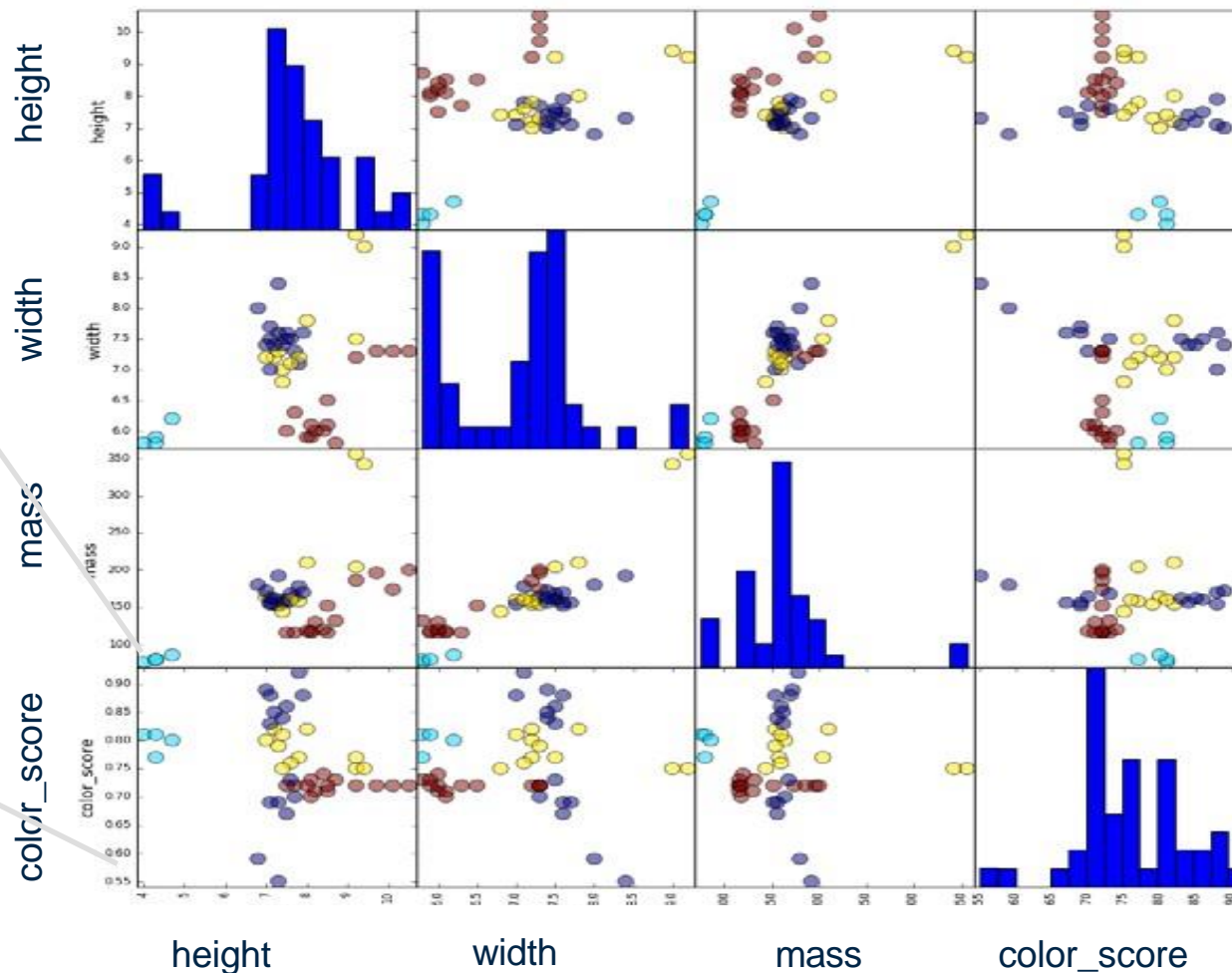
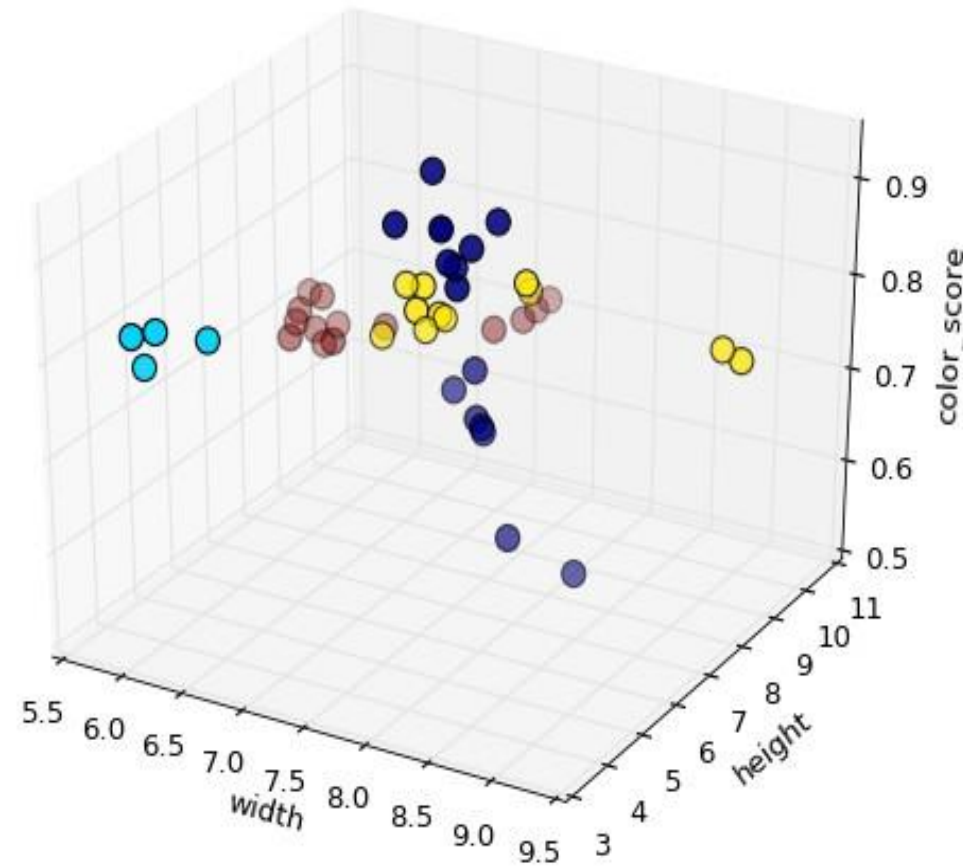


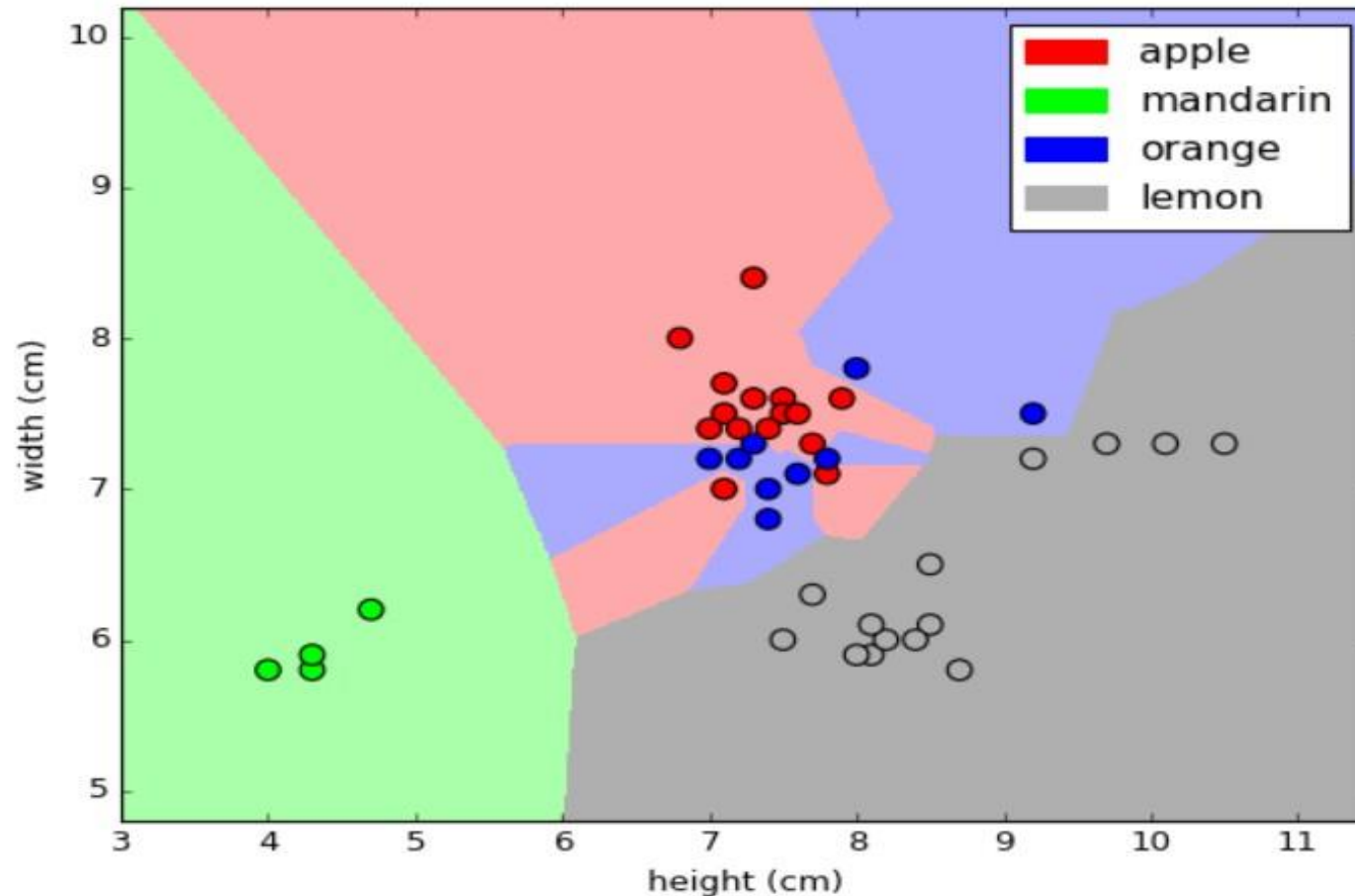
Diagrama de dispersión individual que traza todos los frutos por su altura y color_score. Los colores representan diferentes clases de frutas.



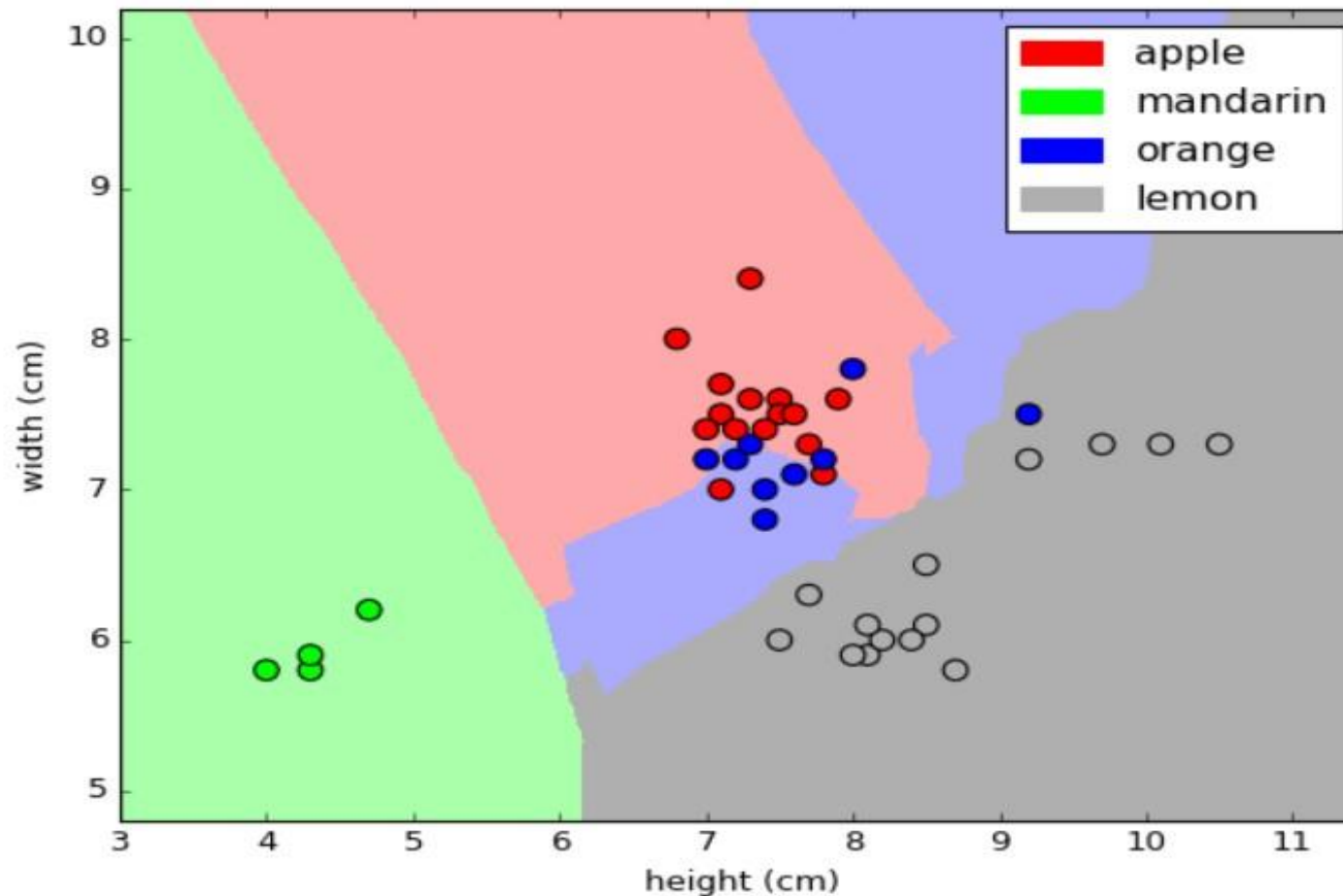
Un diagrama de dispersión de características tridimensionales



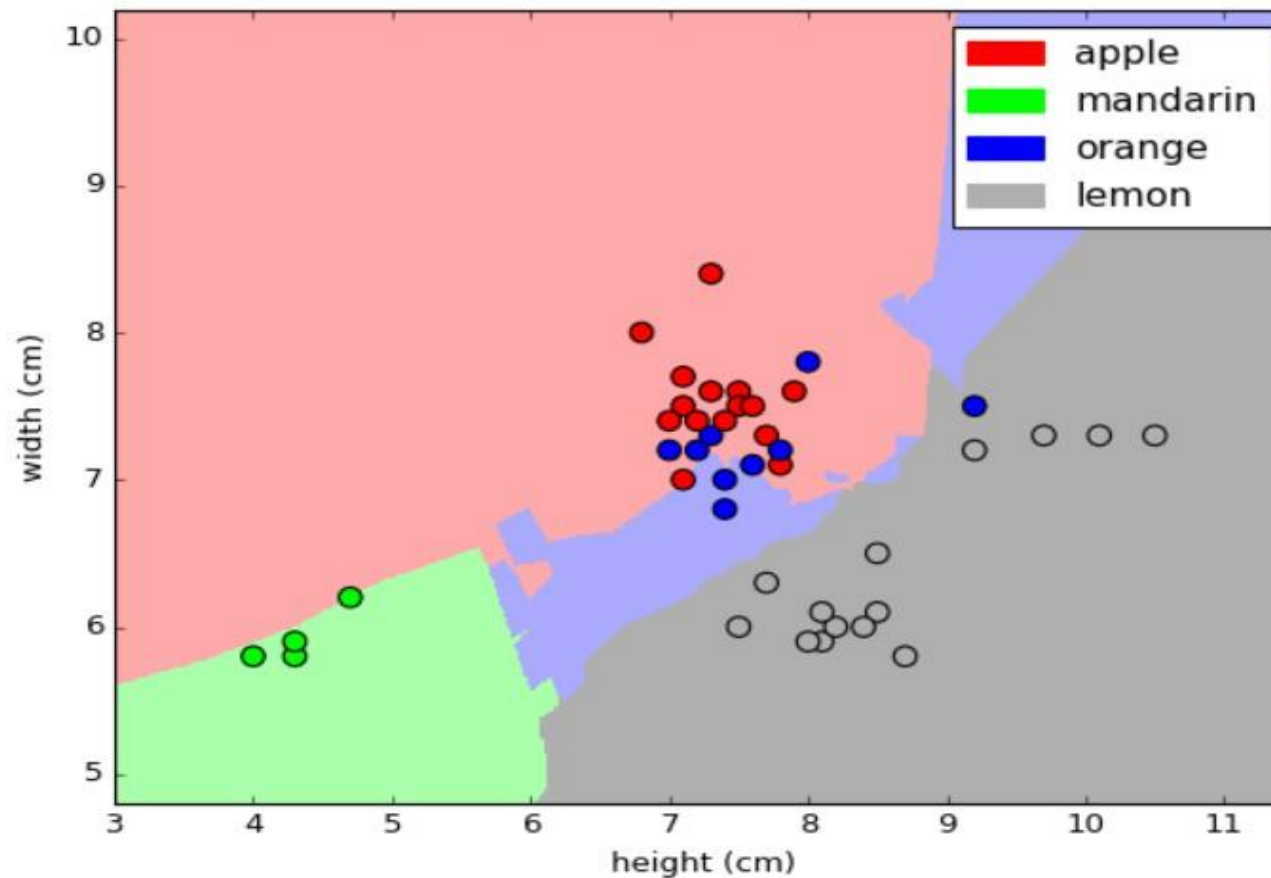
K-vecinos más cercanos ($k=1$) para el conjunto de datos de frutas



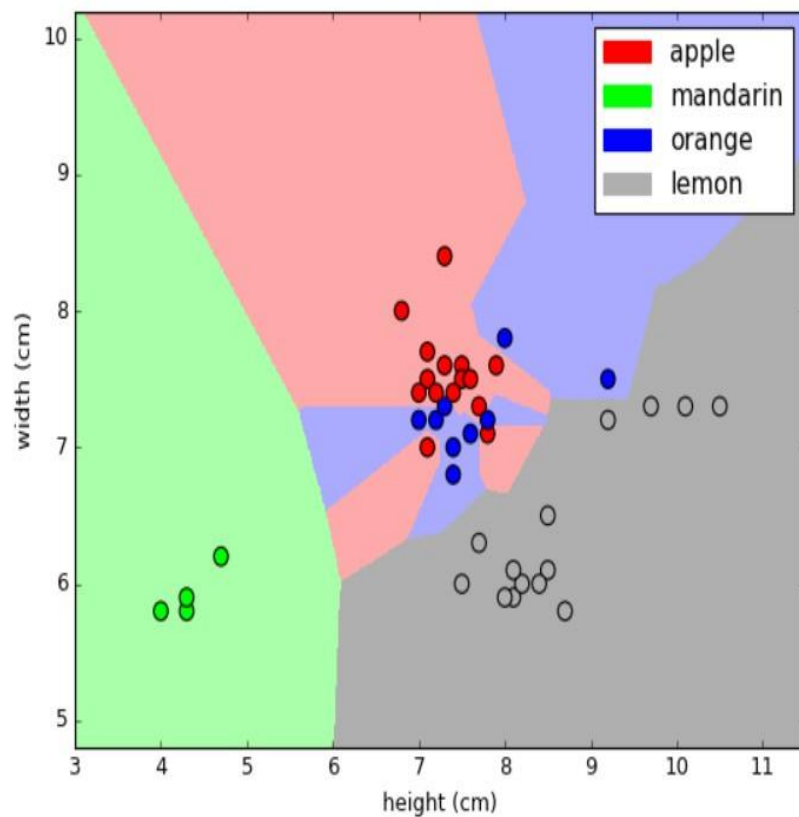
K-vecinos más cercanos ($k=5$) para el conjunto de datos de frutas



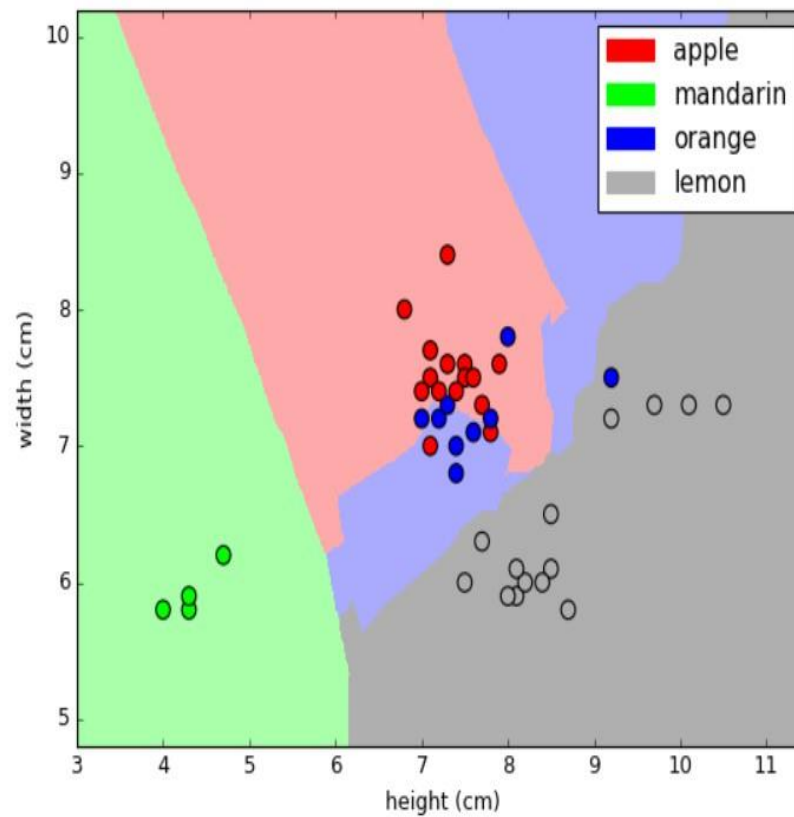
K-vecinos más cercanos ($k=1$) para el conjunto de datos de frutas



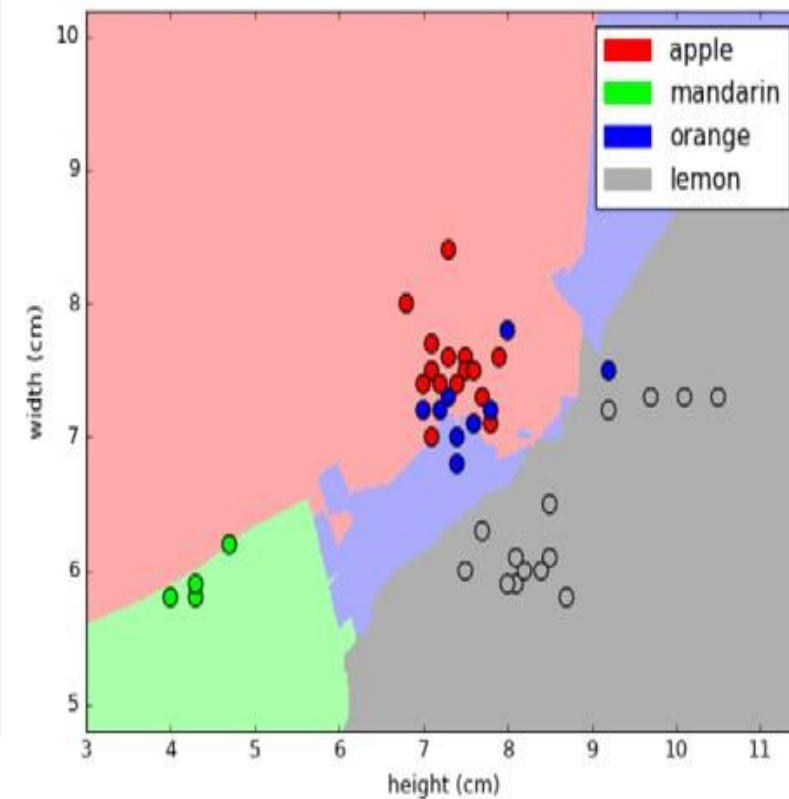
K=1



K=5



K=10



Referencias

Python Intermedio

<https://python-intermedio.readthedocs.io/es/latest/>

Pandas_Cheat_Sheet.

https://pandas.pydata.org/Pandas_Cheat_Sheet.pdf

NearestNeighborsClassification

https://scikit-learn.org/stable/auto_examples/neighbors/plot_classification.html

Confusionmatrix

https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html