Aprendizaje No Supervisado: Clustering

Juan F. Pérez

Departamento MACC
Matemáticas Aplicadas y Ciencias de la Computación
Universidad del Rosario

juanferna.perez@urosario.edu.co

2018

Contenidos

Introducción

Introducción

- Búsqueda de patrones
- Estudio de fenómenos físicos
- Reconocimiento de patrones
- Descubrimiento automático de regularidades
- Algoritmos computacionales

Algunos problemas de aprendizaje

- Datos de entrenamiento contienen las características x_i como las categorías/etiquetas t_i

Algunos problemas de aprendizaje

- Datos de entrenamiento contienen las características x_i como las categorías/etiquetas t_i

Aprendizaje supervisado

Algunos problemas de aprendizaje

- Datos de entrenamiento contienen las características x_i como las categorías/etiquetas t_i

Aprendizaje supervisado

- Resultado es una o varias variables continuas (no un número finito de categorías)

Algunos problemas de aprendizaje

- Datos de entrenamiento contienen las características x_i como las categorías/etiquetas t_i

Aprendizaje supervisado

- Resultado es una o varias variables continuas (no un número finito de categorías)

Regresión

Algunos problemas de aprendizaje

- Datos de entrenamiento contienen las características x_i como las categorías/etiquetas t_i

Algunos problemas de aprendizaje

- Datos de entrenamiento contienen las características x_i como las categorías/etiquetas t_i

Aprendizaje supervisado

Algunos problemas de aprendizaje

- Datos de entrenamiento contienen las características x_i como las categorías/etiquetas t_i

Aprendizaje supervisado

- Resultado es una categoría (de un número finito de posibles categorías)

Algunos problemas de aprendizaje

- Datos de entrenamiento contienen las características x_i como las categorías/etiquetas t_i

Aprendizaje supervisado

- Resultado es una categoría (de un número finito de posibles categorías)

Clasificación

Algunos problemas de aprendizaje

- Datos de entrenamiento contienen las características x_i pero NO las categorías/etiquetas t_i

Algunos problemas de aprendizaje

- Datos de entrenamiento contienen las características x_i pero NO las categorías/etiquetas t_i

Aprendizaje no supervisado

Algunos problemas de aprendizaje

- Datos de entrenamiento contienen las características x_i pero NO las categorías/etiquetas t_i

Aprendizaje no supervisado

- Objetivo es descubrir grupos similares

Algunos problemas de aprendizaje

- Datos de entrenamiento contienen las características x_i pero NO las categorías/etiquetas t_i

Aprendizaje no supervisado

- Objetivo es descubrir grupos similares

Clustering (análisis de conglomerados)

Algunos problemas de aprendizaje

- Datos de entrenamiento contienen las características x_i pero NO las categorías/etiquetas t_i

Algunos problemas de aprendizaje

- Datos de entrenamiento contienen las características x_i pero NO las categorías/etiquetas t_i

Aprendizaje no supervisado

Algunos problemas de aprendizaje

- Datos de entrenamiento contienen las características x_i pero NO las categorías/etiquetas t_i

Aprendizaje no supervisado

- Objetivo es determinar la distribución de los datos en el espacio de entrada

Algunos problemas de aprendizaje

- Datos de entrenamiento contienen las características x_i pero NO las categorías/etiquetas t_i

Aprendizaje no supervisado

- Objetivo es determinar la distribución de los datos en el espacio de entrada

Estimación de densidades

Clustering/Análisis de Conglomerados

Clustering/Análisis de Conglomerados

Objetivo:

- Dadas n observaciones cada una descrita como un vector x de dimensión D (características)

Clustering/Análisis de Conglomerados

Objetivo:

- Dadas n observaciones cada una descrita como un vector x de dimensión D (características)
- Descubrir grupos de observaciones similares

Clustering/Análisis de Conglomerados

Objetivo:

- Dadas n observaciones cada una descrita como un vector x de dimensión D (características)
- Descubrir grupos de observaciones similares
- Grupo = cluster = conglomerado

Clustering/Análisis de Conglomerados

Objetivo:

- Dadas n observaciones cada una descrita como un vector x de dimensión D (características)
- Descubrir grupos de observaciones similares
- Grupo = cluster = conglomerado
- **NO** hay etiquetas $\rightarrow$ no sabemos si efectivamente hay clusters o cuántos hay

Clustering/Análisis de Conglomerados

Objetivo:

- Dadas n observaciones cada una descrita como un vector x de dimensión D (características)
- Descubrir grupos de observaciones similares
- Grupo = cluster = conglomerado
- **NO** hay etiquetas $\rightarrow$ no sabemos si efectivamente hay clusters o cuántos hay
- $\rightarrow$ *diferente* al problema de clasificación

Ejemplos

- Se toman muestras de un tejido canceroso de n pacientes

Ejemplos

- Se toman muestras de un tejido canceroso de n pacientes
- Para cada muestra se obtienen D descriptores (características): medidas físicas, químicas, imágenes

Ejemplos

- Se toman muestras de un tejido canceroso de n pacientes
- Para cada muestra se obtienen D descriptores (características): medidas físicas, químicas, imágenes
- Se busca identificar casos/muestras similares

Ejemplos

- Se toman muestras de un tejido canceroso de n pacientes
- Para cada muestra se obtienen D descriptores (características): medidas físicas, químicas, imágenes
- Se busca identificar casos/muestras similares
- Podrían reflejar estados similares de avance de la enfermedad, respuesta similar a tratamiento, tipos diferentes de enfermedad/paciente

Ejemplos

- Se toman muestras de un tejido canceroso de n pacientes
- Para cada muestra se obtienen D descriptores (características): medidas físicas, químicas, imágenes
- Se busca identificar casos/muestras similares
- Podrían reflejar estados similares de avance de la enfermedad, respuesta similar a tratamiento, tipos diferentes de enfermedad/paciente
- No se sabe *a priori* pero se quiere explorar

Ejemplos

- Se tiene información de n clientes

Ejemplos

- Se tiene información de n clientes
- Para cada cliente se obtienen D descriptores (características): hábitos de compra, datos socio-demográficos

Ejemplos

- Se tiene información de n clientes
- Para cada cliente se obtienen D descriptores (características): hábitos de compra, datos socio-demográficos
- Se busca identificar clientes similares

Ejemplos

- Se tiene información de n clientes
- Para cada cliente se obtienen D descriptores (características): hábitos de compra, datos socio-demográficos
- Se busca identificar clientes similares
- Podrían reflejar potenciales clientes de nuevos productos, interés en ofertas de cierto tipo, capacidad/deseo de compra de ciertos artículos

Ejemplos

- Se tiene información de n clientes
- Para cada cliente se obtienen D descriptores (características): hábitos de compra, datos socio-demográficos
- Se busca identificar clientes similares
- Podrían reflejar potenciales clientes de nuevos productos, interés en ofertas de cierto tipo, capacidad/deseo de compra de ciertos artículos
- No se sabe *a priori* pero se quiere explorar

Ejemplos

- Se tiene información de n clientes
- Para cada cliente se obtienen D descriptores (características): hábitos de compra, datos socio-demográficos
- Se busca identificar clientes similares
- Podrían reflejar potenciales clientes de nuevos productos, interés en ofertas de cierto tipo, capacidad/deseo de compra de ciertos artículos
- No se sabe *a priori* pero se quiere explorar
- (*Segmentación del mercado*)

Resultado

- Clusters de observaciones similares

Resultado

- Clusters de observaciones similares
- Cada cluster puede reflejar un conjunto de interés a analizar

Resultado

- Clusters de observaciones similares
- Cada cluster puede reflejar un conjunto de interés a analizar
- Reducción o simplificación de información

Resultado

- Clusters de observaciones similares
- Cada cluster puede reflejar un conjunto de interés a analizar
- Reducción o simplificación de información
- Simplificar o posibilitar el análisis de grandes cantidades de información multi-dimensional

K-means

K-means

- Agrupar observaciones en K clusters

K-means

- Agrupar observaciones en K clusters
- Para cada observación se determina a cuál de los clusters pertenece (solo uno)

K-means

- Agrupar observaciones en K clusters
- Para cada observación se determina a cuál de los clusters pertenece (solo uno)
- Clusters $C_1, \ldots, C_K$

K-means

- Agrupar observaciones en K clusters
- Para cada observación se determina a cuál de los clusters pertenece (solo uno)
- Clusters $C_1, \ldots, C_K$
- Toda observación pertenece a un solo cluster

K-means

- Visión 1:

K-means

- Visión 1:
  - Observaciones en un mismo cluster deben ser parecidas entre sí

K-means

- Visión 1:
  - Observaciones en un mismo cluster deben ser parecidas entre sí
  - Observaciones en clusters diferentes deben ser relativamente diferentes

K-means

- Visión 1:
  - Observaciones en un mismo cluster deben ser parecidas entre sí
  - Observaciones en clusters diferentes deben ser relativamente diferentes
- Visión 2:

K-means

- Visión 1:
  - Observaciones en un mismo cluster deben ser parecidas entre sí
  - Observaciones en clusters diferentes deben ser relativamente diferentes
- Visión 2:
  - Una observación debe ser más parecida a otras observaciones en el mismo cluster que a observaciones en otros clusters

K-means

- Visión 1:
  - Observaciones en un mismo cluster deben ser parecidas entre sí
  - Observaciones en clusters diferentes deben ser relativamente diferentes
- Visión 2:
  - Una observación debe ser más parecida a otras observaciones en el mismo cluster que a observaciones en otros clusters
- Visión 3:

K-means

- Visión 1:
  - Observaciones en un mismo cluster deben ser parecidas entre sí
  - Observaciones en clusters diferentes deben ser relativamente diferentes
- Visión 2:
  - Una observación debe ser más parecida a otras observaciones en el mismo cluster que a observaciones en otros clusters
- Visión 3:
  - Minimizar la variabilidad al interior de cada cluster conformado

K-means

- Minimizar la variabilidad al interior de cada cluster conformado

K-means

- Minimizar la variabilidad al interior de cada cluster conformado
- Sea V_k una medida de la variabilidad en el cluster k

K-means

- Minimizar la variabilidad al interior de cada cluster conformado
- Sea V_k una medida de la variabilidad en el cluster k
- Objetivo de K-means:

$$\text{mín} \sum_{k=1}^{K} V_k$$

K-means

- Determinar V_k una medidad de la variabilidad en el cluster k

K-means

- Determinar V_k una medidad de la variabilidad en el cluster k
- Distancia (euclideana) entre todos los puntos del cluster

K-means

- Determinar V_k una medidad de la variabilidad en el cluster k
- Distancia (euclideana) entre todos los puntos del cluster
- Suponiendo una sola característica (x_i es un número)

$$V_k = \frac{1}{|C_k|} \sum_{i,j \in C_k} (x_i - x_j)^2$$

K-means

- Determinar V_k una medidad de la variabilidad en el cluster k
- Distancia (euclideana) entre todos los puntos del cluster
- Suponiendo una sola característica (x_i es un número)

$$V_k = \frac{1}{|C_k|} \sum_{i,j \in C_k} (x_i - x_j)^2$$

- Caso general con D características (x_i es un vector con entradas $x_{i,d}$):

$$V_k = \frac{1}{|C_k|} \sum_{i,j \in C_k} \sum_{d=1}^{D} (x_{i,d} - x_{j,d})^2$$

K-means

- Objetivo de K-means:

$$\text{mín} \sum_{k=1}^{K} \frac{1}{|C_k|} \sum_{i,j \in C_k} \sum_{d=1}^{D} (x_{i,d} - x_{j,d})^2$$

K-means

- Objetivo de K-means:

$$\text{mín} \sum_{k=1}^{K} \frac{1}{|C_k|} \sum_{i,j \in C_k} \sum_{d=1}^{D} (x_{i,d} - x_{j,d})^2$$

- Problema de optimización

K-means

- Objetivo de K-means:

$$\text{mín} \sum_{k=1}^{K} \frac{1}{|C_k|} \sum_{i,j \in C_k} \sum_{d=1}^{D} (x_{i,d} - x_{j,d})^2$$

- Problema de optimización
- K^n formas de asignar n observaciones a K clusters

K-means

- Objetivo de K-means:

$$\text{mín} \sum_{k=1}^{K} \frac{1}{|C_k|} \sum_{i,j \in C_k} \sum_{d=1}^{D} (x_{i,d} - x_{j,d})^2$$

- Problema de optimización
- K^n formas de asignar n observaciones a K clusters
- ¿Cómo resolverlo?

K-means: Algoritmo

- Algoritmo sencillo que alcanza encuentra una muy buena solución

K-means: Algoritmo

- Algoritmo sencillo que alcanza encuentra una muy buena solución
- Algoritmo eficiente: ejecución veloz y escalable a muchos datos

K-means: Algoritmo

- Algoritmo sencillo que alcanza encuentra una muy buena solución
- Algoritmo eficiente: ejecución veloz y escalable a muchos datos
- No garantiza que se encuentre la mejor solución (óptimo)

K-means: Algoritmo

- Algoritmo sencillo que alcanza encuentra una muy buena solución
- Algoritmo eficiente: ejecución veloz y escalable a muchos datos
- No garantiza que se encuentre la mejor solución (óptimo)
- El centroide y_k del cluster k:

$$y_k = \frac{1}{|C_k|} \sum_{i \in C_k} \sum_{d=1}^{D} x_{i,d}$$

K-means: Algoritmo

- Algoritmo sencillo que alcanza encuentra una muy buena solución
- Algoritmo eficiente: ejecución veloz y escalable a muchos datos
- No garantiza que se encuentre la mejor solución (óptimo)
- El centroide y_k del cluster k:

$$y_k = \frac{1}{|C_k|} \sum_{i \in C_k} \sum_{d=1}^{D} x_{i,d}$$

- Centroide y_k: punto medio del cluster k

K-means: Algoritmo

1. Seleccione K centroides al azar, uno para cada cluster
2. Hasta que los centroides no cambien más, *repita*:
   1) Asigne cada observación al cluster más cercano
   2) Para cada cluster, re-calcule el centroide

K-Means en Scikit-Learn

K-Means en Scikit-Learn

```
import matplotlib.pyplot as plt
import seaborn as sns; sns.set()

from sklearn.datasets.samples_generator
            import make_blobs
X, y = make_blobs(n_samples=300, centers=4,
            cluster_std=0.6, random_state=1)
plt.scatter(X[:,0], X[:,1], s=50)
```

K-Means en Scikit-Learn

```
from sklearn.cluster import KMeans
kmeans = KMeans(n_clusters=4)
y_kmeans = kmeans.fit_predict(X)

plt.scatter(X[:,0], X[:,1], c=y_kmeans, s=50,
            cmap='viridis')
centros = kmeans.cluster_centers_
print(centros)
plt.scatter(centros[:,0], centros[:,1],
            c='black', s=200, alpha=0.5)
```

K-Means en Scikit-Learn: Evolución

```
from sklearn.datasets.samples_generator
            import make_blobs

X, y = make_blobs(n_samples=300, centers=4,
            cluster_std=0.6, random_state=1)
plt.figure()
plt.scatter(X[:,0], X[:,1], c='green', s=50)

cents, etiqs = kmeans_paso_a_paso(X,4,3,100)
```

K-Means en Scikit-Learn: Evolución

```
from sklearn.datasets.samples_generator
            import make_blobs

X, y = make_blobs(n_samples=300, centers=4,
            cluster_std=0.6, random_state=1)
plt.figure()
plt.scatter(X[:,0], X[:,1], c='green', s=50)

cents, etiqs = kmeans_paso_a_paso(X,4,3,100)
```

- Descargar la función `kmeans_paso_a_paso` del repositorio

K-Means para reconocer dígitos similares

```
import matplotlib.pyplot as plt
import seaborn as sns; sns.set()
import numpy as np
from sklearn.datasets import load_digits
from sklearn.cluster import KMeans
```

K-Means para reconocer dígitos similares

```
digits = load_digits()
print(digits.data.shape)

kmeans = KMeans(n_clusters=10, random_state=0)
clusters = kmeans.fit_predict(digits.data)
kmeans.cluster_centers_.shape
```

K-Means para reconocer dígitos similares

```
fig, ax = plt.subplots(2, 5, figsize=(8, 3))
centers = kmeans.cluster_centers_.reshape(10, 8, 8)
for axi, center in zip(ax.flat, centers):
    axi.set(xticks=[], yticks=[])
    axi.imshow(center, interpolation='nearest',
               cmap=plt.cm.binary)
```

K-Means para reconocer dígitos similares

```
from scipy.stats import mode
etiqs = np.zeros_like(clusters)
for i in range(10):
    mask = (clusters == i)
    etiqs[mask] = mode(digits.target[mask])[0]
```

K-Means para reconocer dígitos similares

```
from sklearn.metrics import confusion_matrix
mat = confusion_matrix(digits.target, etiqs)
plt.figure()
sns.heatmap(mat.T, square=True, annot=True, fmt='d',
            cbar=False,
            xticklabels=digits.target_names,
            yticklabels=digits.target_names)
plt.xlabel('etiqueta observada')
plt.ylabel('etiqueta predicha')
```

K-Means para reconocer dígitos similares

```
from sklearn.metrics import accuracy_score
print("precisión:",
      accuracy_score(digits.target, etiqs))
```

K-Means para comprimir imágenes

K-Means para comprimir imágenes

```
from sklearn.datasets import load_sample_image
import matplotlib.pyplot as plt
import numpy as np

china = load_sample_image("china.jpg")
ax = plt.axes(xticks=[], yticks=[])
ax.imshow(china)
```

K-Means para comprimir imágenes

```
print(china.shape)
data = china / 255.0 # escala 0 a 1
data = data.reshape(427 * 640, 3)
print(data.shape)
```

K-Means para comprimir imágenes

```
def plot_pixels(data, title, colors=None, N=10000):
    if colors is None:
        colors = data

    # seleccionar sub-conjunto aleatoriamente
    rng = np.random.RandomState(0)
    i = rng.permutation(data.shape[0])[:N]
    colors = colors[i]
```

K-Means para comprimir imágenes

```
R, G, B = data[i].T
fig, ax = plt.subplots(1, 2, figsize=(16, 6))
ax[0].scatter(R, G, color=colors, marker='.')
ax[0].set(xlabel='Red', ylabel='Green',
                    xlim=(0, 1), ylim=(0, 1))
ax[1].scatter(R, B, color=colors, marker='.')
ax[1].set(xlabel='Red', ylabel='Blue',
                    xlim=(0, 1), ylim=(0, 1))
fig.suptitle(title, size=20);
```

K-Means para comprimir imágenes

```
plot_pixels(data, title='Espacio de Colores inicial:
16 millones de colores posibles')
```

K-Means para comprimir imágenes

```
from sklearn.cluster import MiniBatchKMeans
kmeans = MiniBatchKMeans(16)
kmeans.fit(data)
new_colors = kmeans.cluster_centers_[
        kmeans.predict(data)]
plot_pixels(data, colors=new_colors,
        title="Espacio reducido: 16 colores")
```

K-Means para comprimir imágenes

```
china_recolored = new_colors.reshape(china.shape)
plt.figure()
fig, ax = plt.subplots(1, 2, figsize=(16, 6),
                       subplot_kw=dict(xticks=[], yticks=[]))
```

K-Means para comprimir imágenes

```
fig.subplots_adjust(wspace=0.05)
ax[0].imshow(china)
ax[0].set_title('Imagen original', size=16)
ax[1].imshow(china_recolored)
ax[1].set_title('Imagen en 16 colores', size=16)
```

Datos sin fronteras lineales

Datos sin fronteras lineales

```
import matplotlib.pyplot as plt
import seaborn as sns; sns.set()
import numpy as np

from sklearn.datasets import make_moons
X, y = make_moons(n_samples=300, noise = 0.05,
                  random_state=1)
plt.scatter(X[:,0], X[:,1], s=50)
```

Datos sin fronteras lineales

```
from sklearn.cluster import KMeans
kmeans = KMeans(n_clusters=2)
kmeans.fit(X)
y_kmeans = kmeans.predict(X)

plt.scatter(X[:,0], X[:,1], c=y_kmeans, s=50,
            cmap='viridis')
centers = kmeans.cluster_centers_
print(centers)
plt.scatter(centers[:,0], centers[:,1], c='black',
            s=200, alpha=0.5 )
```

Datos sin fronteras lineales

```
from sklearn.cluster import SpectralClustering
modelo = SpectralClustering(n_clusters=2,
                            affinity='nearest_neighbors',
                            assign_labels='kmeans')
y_modelo = modelo.fit_predict(X)

plt.figure()
plt.scatter(X[:,0], X[:,1], c=y_modelo, s=50,
                          cmap='viridis')
```

Clustering Jerárquico

Clustering Jerárquico

Dendogramas

```
import numpy as np

X = np.array([[5,3],
     [10,15],
     [15,12],
     [24,10],
     [30,30],
     [85,70],
     [71,80],
     [60,78],
     [70,55],
     [80,91],])
```

Clustering Jerárquico

Dendogramas y Clustering Jerárquico

```
import matplotlib.pyplot as plt
import seaborn as sns; sns.set()
import numpy as np

from sklearn.datasets.samples_generator import make_bl
X, y = make_blobs(n_samples=300, centers=4,
                  cluster_std=0.6, random_state=1)
plt.scatter(X[:,0], X[:,1], s=50)
```

Clustering Jerárquico

```
import scipy.cluster.hierarchy as shc
plt.figure()
plt.title(u"Dendograma de característica 1")
dend = shc.dendrogram(shc.linkage(X, method='ward'))
```

Clustering Jerárquico

```
from sklearn.cluster import AgglomerativeClustering
modelo = AgglomerativeClustering(n_clusters=4,
         affinity='euclidean',linkage='ward')
y_modelo = modelo.fit_predict(X)

plt.figure()
plt.scatter(X[:,0], X[:,1], c=y_modelo, s=50,
         cmap='viridis')
```