

# Üzleti Elemzések Módszertana

## 7. Előadás: Felügyelet nélküli tanulás

Kuknyó Dániel  
Budapesti Gazdasági Egyetem

2023/24  
2. félév

Dimenzionalitás  
○●○○○

Manifold tanulás  
○○○○

Dimenziócsökkentés  
○○○○○○

Klaszteranalízis  
○○○○○○○○○○

DBSCAN  
○○○

## 1 Dimenzionalitás

## 2 Manifold tanulás

## 3 Dimenziócsökkentés

## 4 Klaszteranalízis

## 5 DBSCAN

Dimenzionalitás  
oo●oo

Manifold tanulás  
oooo

Dimenziócsökkentés  
oooooo

Klaszteranalízis  
oooooooo

DBSCAN  
ooo

## 1 Dimenzionalitás

## 2 Manifold tanulás

## 3 Dimenziócsökkentés

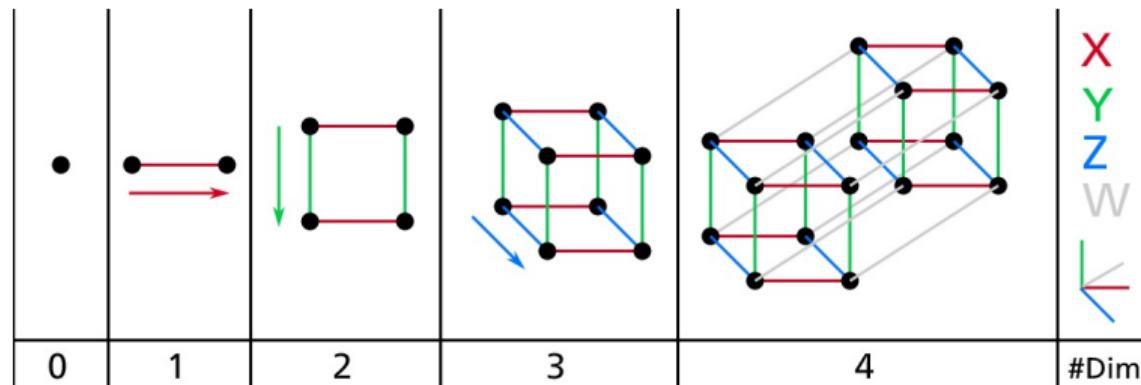
## 4 Klaszteranalízis

## 5 DBSCAN

# A dimenzionalitás átka

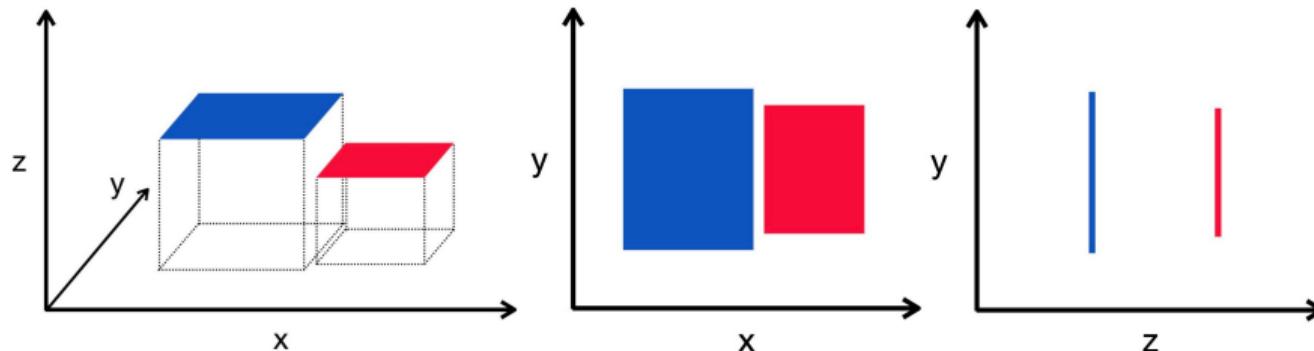
Az érzékek gyakran megcsalhatják az embereket, mert 3D-be vannak zárva. A matematikában viszont lehetséges számolni magasabb dimenziókkal. Sok koncepció nagyon különbözően viselkedik magasabb dimenzióban mint alacsonyabban.

Például: egy egység négyzetben az a valószínűség, hogy egy véletlen pont 0.001-nél közelebb lesz egy élhez, 0.4%. Ugyanez a valószínűség egy 10000 dimenziós hiperkocka esetén 99.99999%.



# Dimenziócsökkentés

Olyan eljárások, amelyek egy **magasabb dimenziós térből** egy **alacsonyabb dimenziós térbe** transzformálják az adatokat amellett, hogy a fontos információtartalom ne vesszen el. Főbb eljárásai a jellemzőkiválasztás, jellemző összevonás, manifold tanulás, izometrikus leképezés.



Dimenzionalitás  
ooooo

Manifold tanulás  
●ooo

Dimenziócsökkentés  
oooooo

Klaszteranalízis  
oooooooo

DBSCAN  
ooo

## 1 Dimenzionalitás

## 2 Manifold tanulás

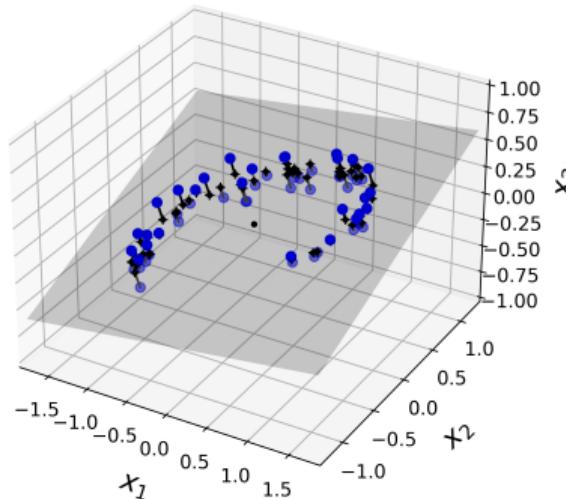
## 3 Dimenziócsökkentés

## 4 Klaszteranalízis

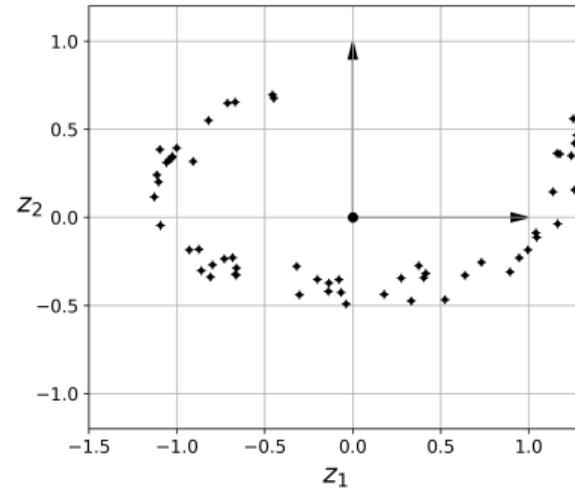
## 5 DBSCAN

# Projekció

A projekció, mint dimenziócsökkentési eljárás egy **adathalmaz jellemzőinek csökkentett dimenziójú térbe való leképezését jelenti.**



Ez a folyamat lehetővé teszi a magas dimenziójú adatok egyszerűbb, kezelhetőbb formában való vizsgálatát.



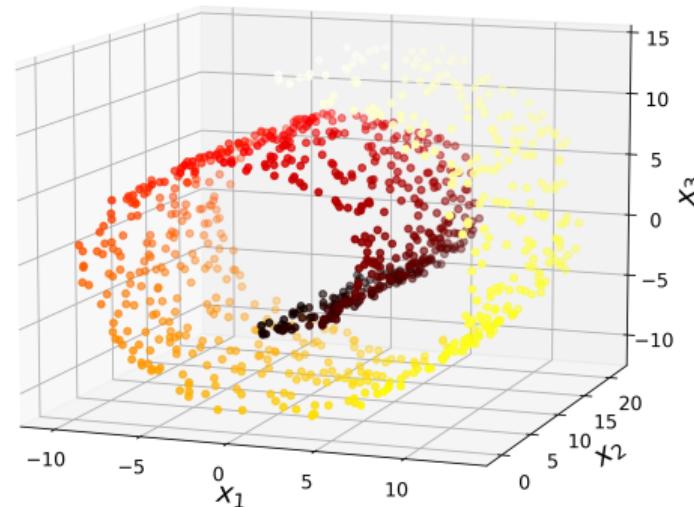
# A svéd tekercs

A svéd tekercs egy gyakran használt tesztelő adathalmaz a manifold tanulás területén.

A manifold tanulás célja a **magas dimenziós adatok alacsony dimenziós, de topologikusan hű reprezentációjának megtalálása**. A svéd tekercs egy 3D-ben lévő 2D-s manifold, amely két dimenzióban van tekerve vagy spirálozva.

A feladat a tekercs lehajtása a belső struktúrájának megőrzésével.

Svéd tekercs

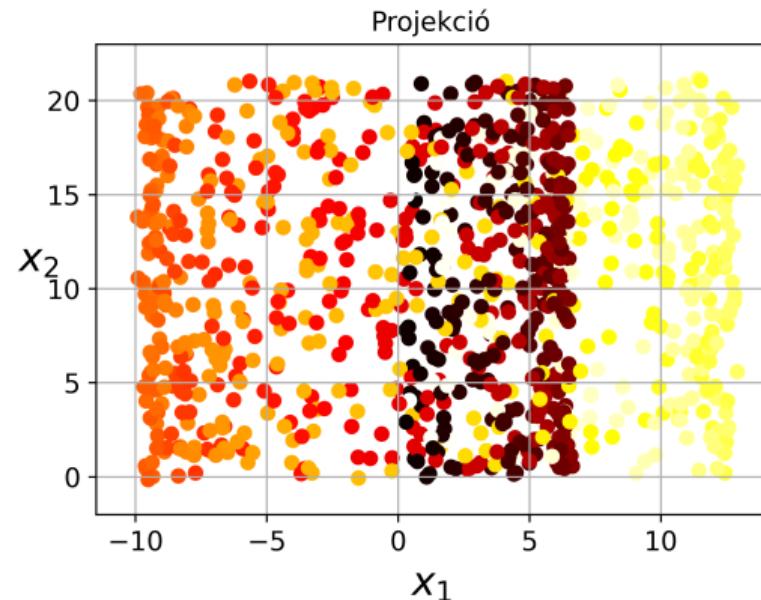


# A svéd tekercs

A svéd tekercs egy gyakran használt tesztelő adathalmaz a manifold tanulás területén.

A manifold tanulás célja a **magas dimenziós adatok alacsony dimenziós, de topologikusan hű reprezentációjának megtalálása**. A svéd tekercs egy 3D-ben lévő 2D-s manifold, amely két dimenzióban van tekerve vagy spirálozva.

A feladat a tekercs lehajtása a belső struktúrájának megőrzésével.

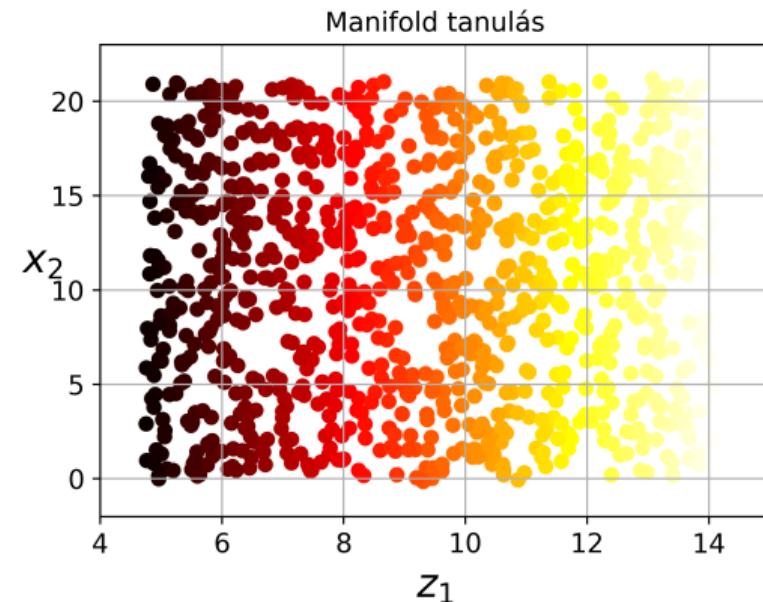


# A svéd tekercs

A svéd tekercs egy gyakran használt tesztelő adathalmaz a manifold tanulás területén.

A manifold tanulás célja a **magas dimenziós adatok alacsony dimenziós, de topologikusan hű reprezentációjának megtalálása**. A svéd tekercs egy 3D-ben lévő 2D-s manifold, amely két dimenzióban van tekerve vagy spirálozva.

A feladat a tekercs lehajtása a belső struktúrájának megőrzésével.



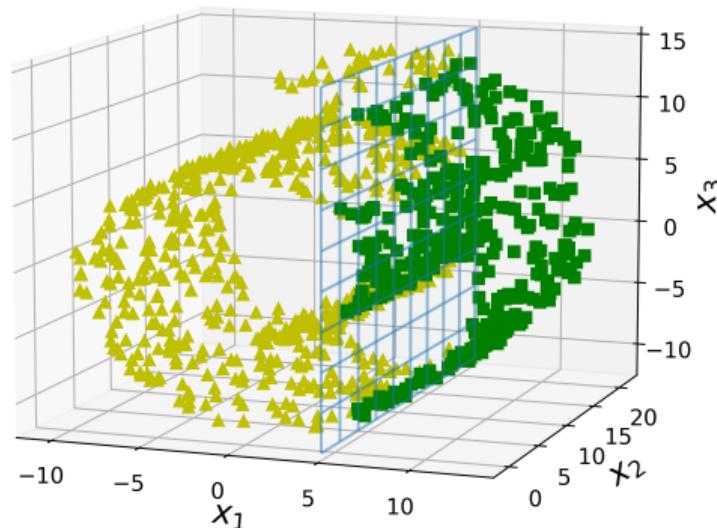
# A manifold hipotézis

## Manifold hipotézis

A manifold feltevés két alapvetően fontos dolgot állít:

- Az adathalmazban **rejlenek** manifoldok. Ez leggyakrabban empírikusan megfigyelhető.
- A feladat **könnyebb** lesz, ha a manifoldnak egy alacsonyabb dimenziós terében próbáljuk megoldani.

A manifold tanulás nem minden esetben alkalmazható.



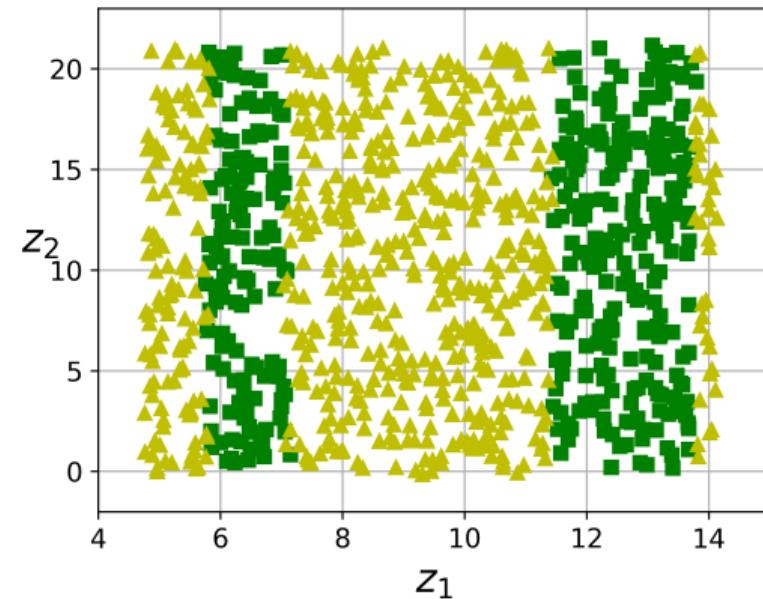
# A manifold hipotézis

## Manifold hipotézis

A manifold feltevés két alapvetően fontos dolgot állít:

- Az adathalmazban **rejlenek** manifoldok. Ez leggyakrabban empírikusan megfigyelhető.
- A feladat **könnyebb** lesz, ha a manifoldnak egy alacsonyabb dimenziós terében próbáljuk megoldani.

A manifold tanulás nem minden esetben alkalmazható.



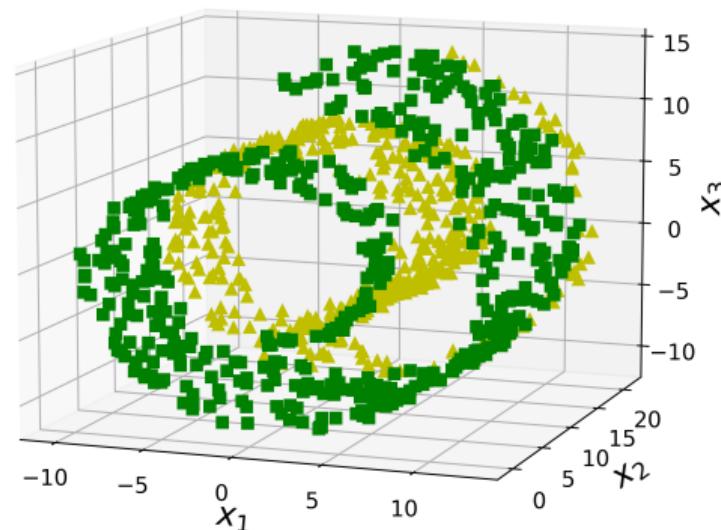
# A manifold hipotézis

## Manifold hipotézis

A manifold feltevés két alapvetően fontos dolgot állít:

- Az adathalmazban **rejlenek** manifoldok. Ez leggyakrabban empírikusan megfigyelhető.
- A feladat **könnyebb** lesz, ha a manifoldnak egy alacsonyabb dimenziós terében próbáljuk megoldani.

A manifold tanulás nem minden esetben alkalmazható.



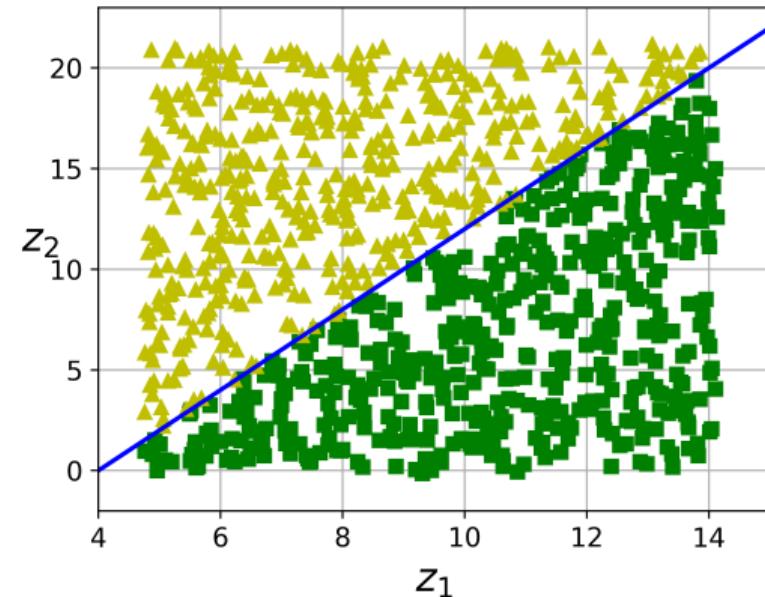
# A manifold hipotézis

## Manifold hipotézis

A manifold feltevés két alapvetően fontos dolgot állít:

- Az adathalmazban **rejlenek** manifoldok. Ez leggyakrabban empírikusan megfigyelhető.
- A feladat **könnyebb** lesz, ha a manifoldnak egy alacsonyabb dimenziós terében próbáljuk megoldani.

A manifold tanulás nem minden esetben alkalmazható.



Dimenzionalitás  
ooooo

Manifold tanulás  
oooo

Dimenziócsökkentés  
●ooooo

Klaszteranalízis  
oooooooo

DBSCAN  
ooo

## 1 Dimenzionalitás

## 2 Manifold tanulás

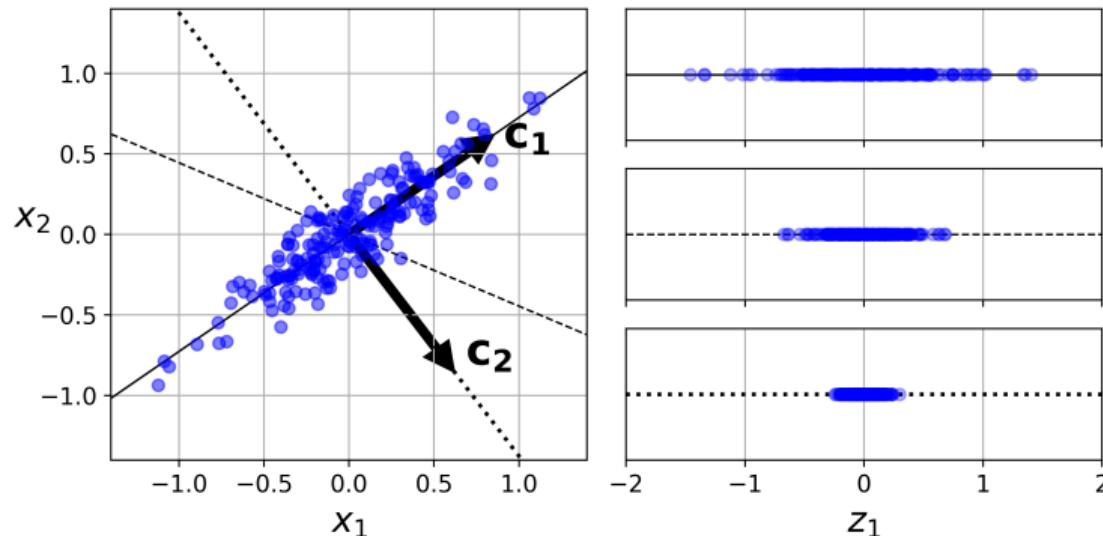
## 3 Dimenziócsökkentés

## 4 Klaszteranalízis

## 5 DBSCAN

# Főkomponenselemzés

Az egyik legnépszerűbb dimenziócsökkentő algoritmus. Először felfedezi azt a hipersíket, amely a legközelebb van az adatokhoz, majd ráprojektálja az adatpontokat. Az eljárás maximalizálja az eredeti adathalmaz és a projekciója közötti varianciát, hiszen az információ a varianciában rejlik.



# Szinguláris értékfelbontás

## Szinguláris értékfelbontás

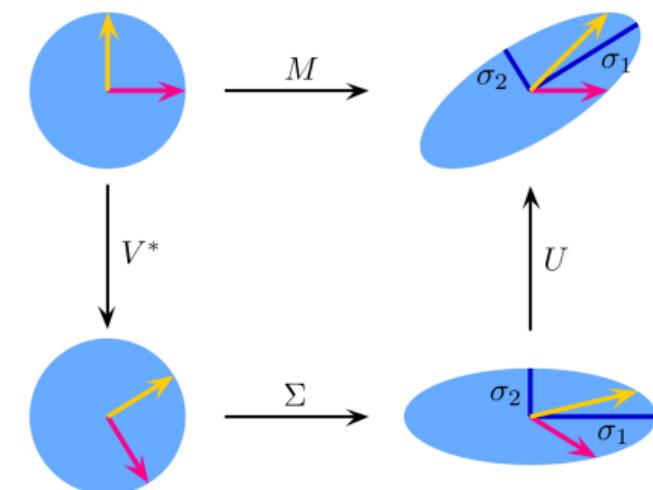
Legyen az egységvektor, amelyik az  $i$ -edik tengelyt definiálja, az  $i$ -edik főkomponens:  $c_1, c_2, \dots, c_n$ .

Az eljárás  $M \in \mathbb{R}$  mátrixot három mátrixra bontja  $M = U\Sigma V^T$ :

- $U$ :  $n$  db ortonormált sajátvektort tartalmaz
- $\Sigma$ : Főátlóján találhatóak a szinguláris értékek
- $V$ : A keresett főkomponenseket tartalmazza

A főkomponensek létrejötte után a projekcióhoz  $M$  tanító adathalmaz mátrix és  $W_d$  főkomponenseket tartalmazó mátrix szorzatát kell képezni:

$$M_{d\text{-proj}} = M \cdot W_d.$$



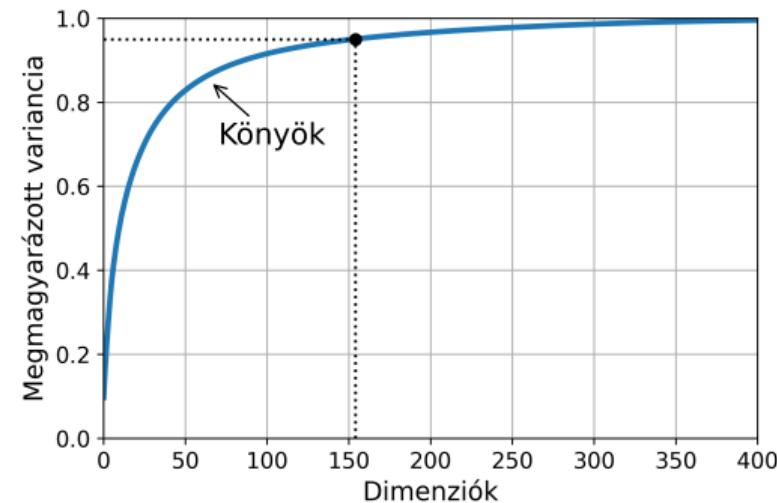
$$M = U \cdot \Sigma \cdot V^*$$

# A megfelelő dimenziószám kiválasztása

## A dimenziók száma a PCA

hiperparamétere, ezért létrejöttek eljárások a finomhangolására. Az egyik ilyen eljárás a könyök módszer:

A könyök módszer szerint ahol elkezd csökkenni a modell által megmagyarázott variancia növekedési üteme, ott könyökpont található, nem kell több dimenziót bevinni a modellezésbe.

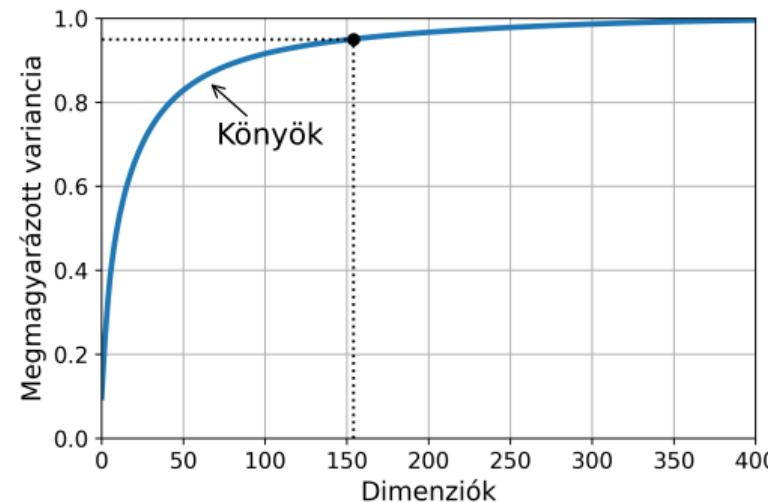


# A megfelelő dimenziószám kiválasztása

## Megmagyarázott varianciahányad

Megadja, hogy az egyes főkomponensek, vagy dimenziók, **milyen mértékben járulnak hozzá** az eredeti adathalmaz **varianciájának magyarázatához**.

A főkomponensek varianciájának és az összes varianciának a hányadosa.



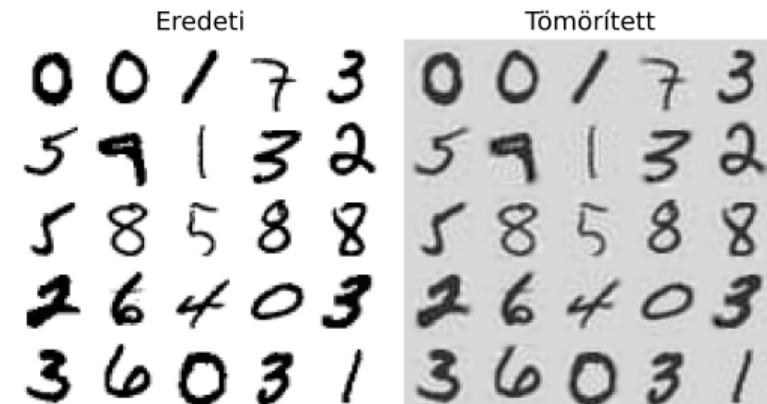
# PCA, mint tömörítési eljárás

Dimenziócsökkentés után az adatkészlet kevesebb memóriát foglal.

Az MNIST adathalmazban 95% variancia megtartással 28x28 képek esetén 784 jellemzőből 150 maradt. Ez több, mint 80% csökkenés 5% varianciáért cserébe.

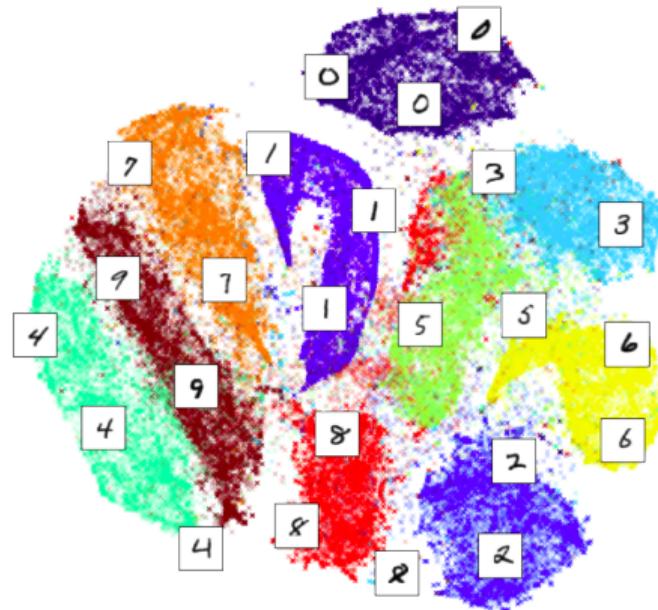
A transzformáció fordított irányba is lehetséges:

$$X_{recovered} = X_{d-proj} W_d^T$$



# Dimenziócsökkentés az MNIST adatokon

A képen az MNIST adathalmaz látható T-SNE (T-eloszlású sztochasztikus szomszéd beágyazás) modell által 2 dimenzióba projektálva.



Dimenzionalitás  
ooooo

Manifold tanulás  
oooo

Dimenziócsökkentés  
oooooo

Klaszteranalízis  
●oooooooo

DBSCAN  
ooo

## 1 Dimenzionalitás

## 2 Manifold tanulás

## 3 Dimenziócsökkentés

## 4 Klaszteranalízis

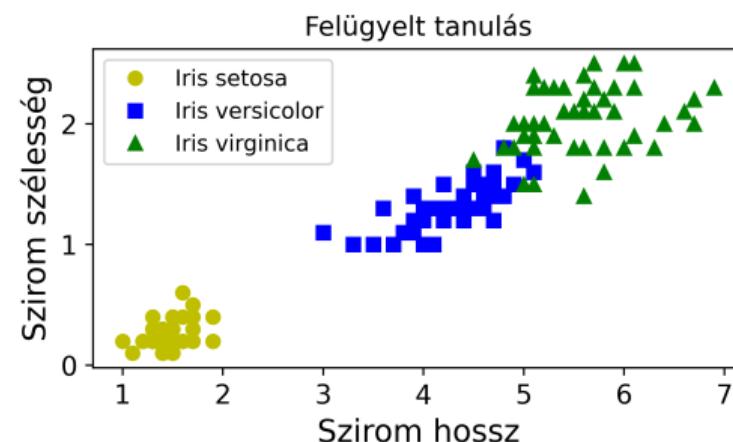
## 5 DBSCAN

# Klaszteranalízis

A klaszteranalízis feladata **felismerni az egymással hasonló egyedekeket, majd csoportokba (klaszterekbe) rendezni őket.**

A klaszterezés egy **felügyelet nélküli tanulási módszer**, tehát az adatok címkéi nem ismertek a tanítási folyamat során, a modell feladata ezeket megtalálni. Főbb felhasználásai:

- Vásárló szegmentálás
- Új adatok elemzése
- Dimenziócsökkentés
- Anomália detekció
- Képek szegmentálása

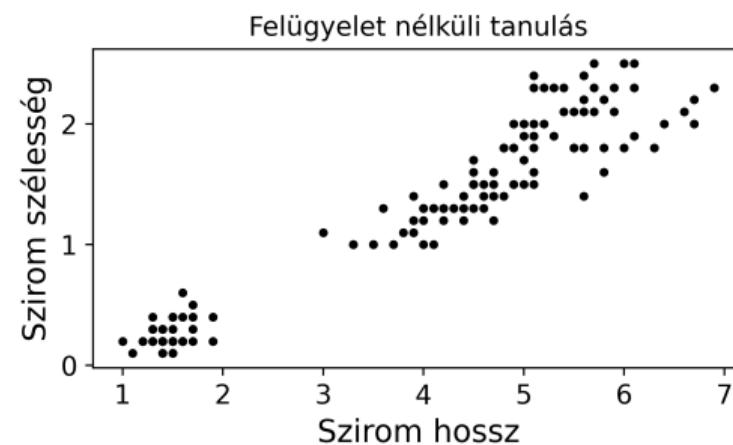


# Klaszteranalízis

A klaszteranalízis feladata **felismerni az egymással hasonló egyedeket, majd csoportokba (klaszterekbe) rendezni őket.**

A klaszterezés egy **felügyelet nélküli tanulási módszer**, tehát az adatok címkéi nem ismertek a tanítási folyamat során, a modell feladata ezeket megtalálni. Főbb felhasználásai:

- Vásárló szegmentálás
- Új adatok elemzése
- Dimenziócsökkentés
- Anomália detekció
- Képek szegmentálása

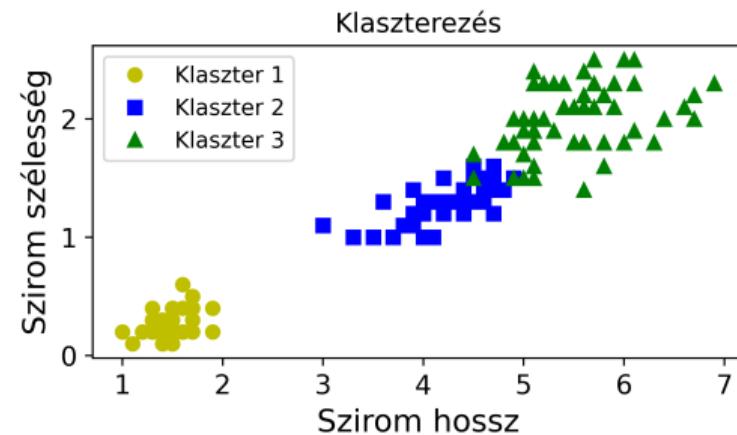


# Klaszteranalízis

A klaszteranalízis feladata **felismerni az egymással hasonló egyedeket, majd csoportokba (klaszterekbe) rendezni őket.**

A klaszterezés egy **felügyelet nélküli tanulási módszer**, tehát az adatok címkéi nem ismertek a tanítási folyamat során, a modell feladata ezeket megtalálni. Főbb felhasználásai:

- Vásárló szegmentálás
- Új adatok elemzése
- Dimenziócsökkentés
- Anomália detekció
- Képek szegmentálása

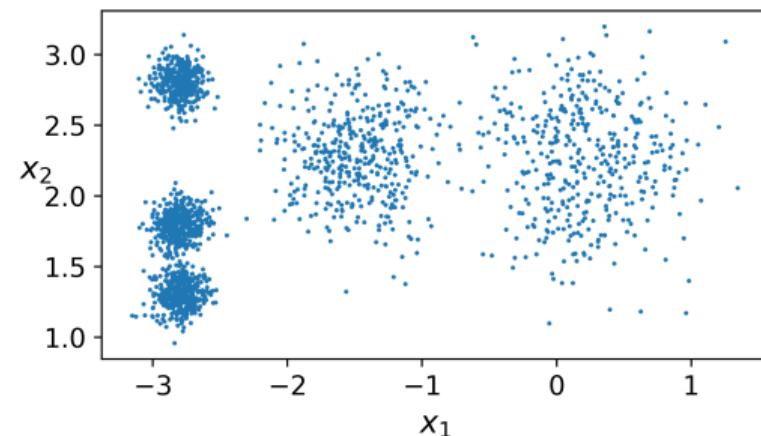


# A $K$ -közép algoritmus

A  $K$  közép algoritmus célja megkeresni minden csoport középpontját, majd minden adatpontot a hozzá legközelebb eső középponthoz rendel. Ez a középpont a **centroid**.

A létrehozandó klaszterek száma  $K$  az algoritmus hiperparamétere, tehát a tanítónak kell megadnia a futtatás előtt.

A létrejövő döntési határok egy **Voronoi-tesszellációt** határoznak meg.

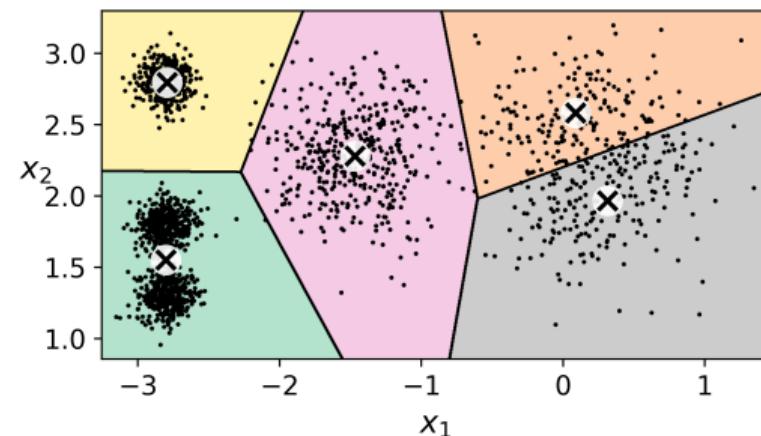


# A $K$ -közép algoritmus

A  $K$  közép algoritmus célja megkeresni minden csoport középpontját, majd minden adatpontot a hozzá legközelebb eső középponthoz rendel. Ez a középpont a **centroid**.

A létrehozandó klaszterek száma  $K$  az algoritmus hiperparamétere, tehát a tanítónak kell megadnia a futtatás előtt.

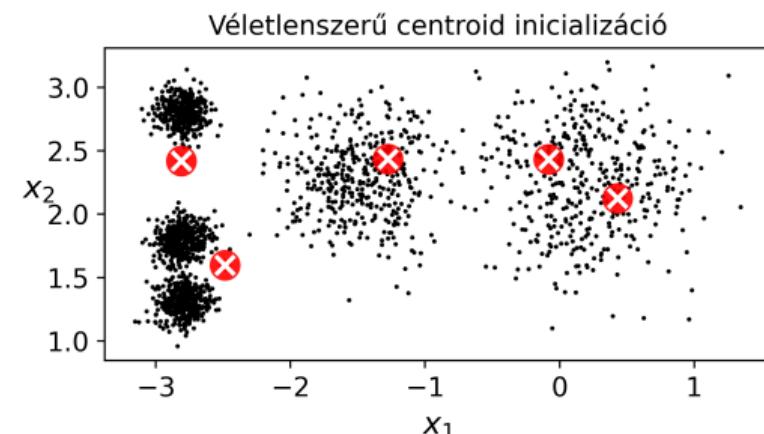
A létrejövő döntési határok egy **Voronoi-tesszellációt** határoznak meg.



# A $K$ -közép működése

Az algoritmus két lépést ismétel, ameddig a klaszter centroidok mozgása nem lassul le a kilépési határ alá:

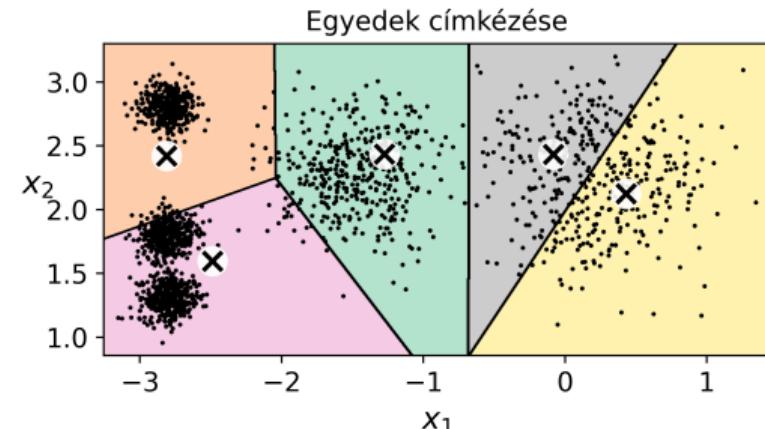
- ① Véletlenszerű centroid inicializáció
- ② minden egyed hozzárendelése a hozzá legközelebb eső centroidhez
- ③ minden centroid mozgatása a hozzá tartozó egyedek várható értékére



# A $K$ -közép működése

Az algoritmus két lépést ismétel, ameddig a klaszter centroidok mozgása nem lassul le a kilépési határ alá:

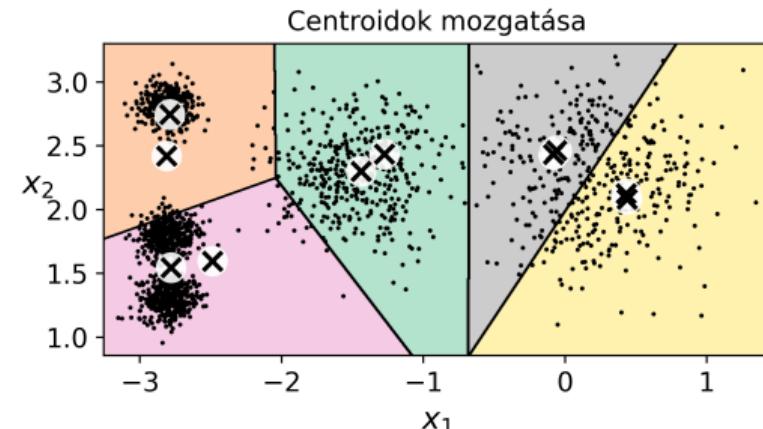
- ① Véletlenszerű centroid inicializáció
- ② minden egyed hozzárendelése a hozzá legközelebb eső centroidhez
- ③ minden centroid mozgatása a hozzá tartozó egyedek várható értékére



# A $K$ -közép működése

Az algoritmus két lépést ismétel, ameddig a klaszter centroidok mozgása nem lassul le a kilépési határ alá:

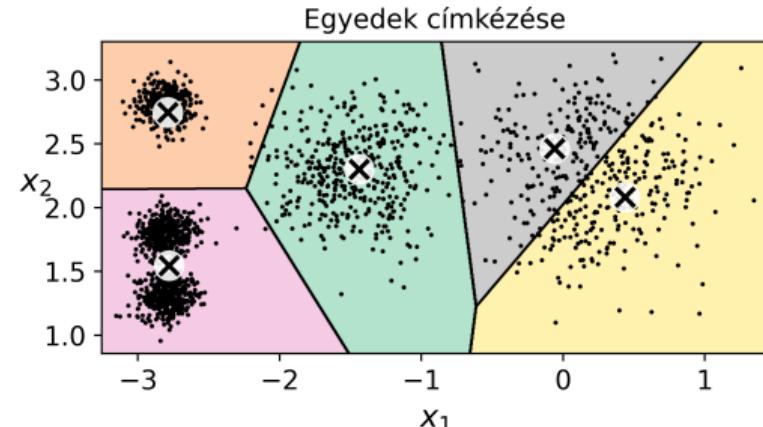
- ① Véletlenszerű centroid inicializáció
- ② minden egyed hozzárendelése a hozzá legközelebb eső centroidhez
- ③ minden centroid mozgatása a hozzá tartozó egyedek várható értékére



# A $K$ -közép működése

Az algoritmus két lépést ismétel, ameddig a klaszter centroidok mozgása nem lassul le a kilépési határ alá:

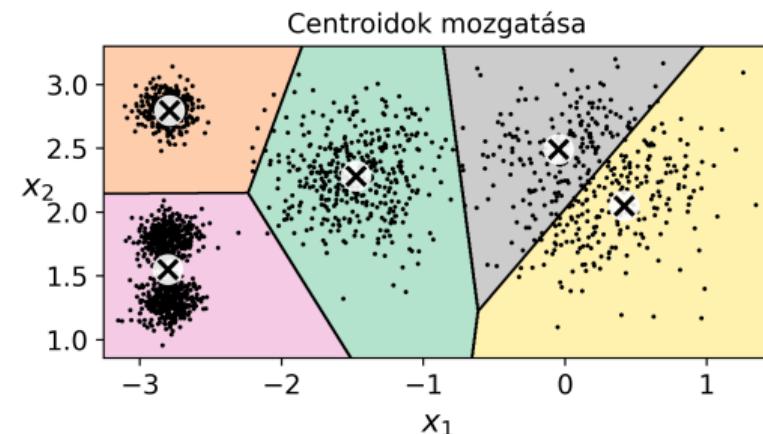
- ① Véletlenszerű centroid inicializáció
- ② minden egyed hozzárendelése a hozzá legközelebb eső centroidhez
- ③ minden centroid mozgatása a hozzá tartozó egyedek várható értékére



# A $K$ -közép működése

Az algoritmus két lépést ismétel, ameddig a klaszter centroidek mozgása nem lassul le a kilépési határ alá:

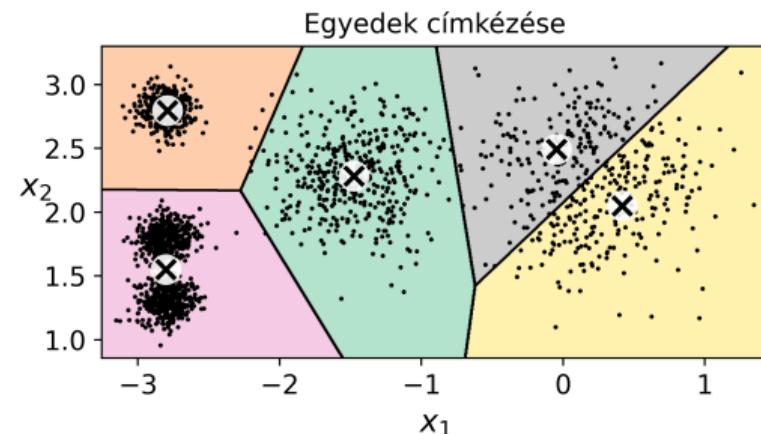
- ① Véletlenszerű centroid inicializáció
- ② minden egyed hozzárendelése a hozzá legközelebb eső centroidhez
- ③ minden centroid mozgatása a hozzá tartozó egyedek várható értékére



# A $K$ -közép működése

Az algoritmus két lépést ismétel, ameddig a klaszter centroidok mozgása nem lassul le a kilépési határ alá:

- ① Véletlenszerű centroid inicializáció
- ② minden egyed hozzárendelése a hozzá legközelebb eső centroidhez
- ③ minden centroid mozgatása a hozzá tartozó egyedek várható értékére



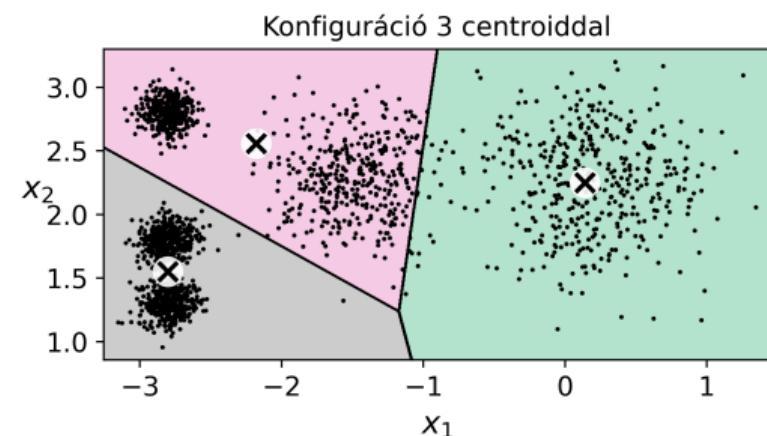
# Optimális klaszterszám megtalálása

A korábbi példában  $K = 5$  volt a klaszterszám, mert ez az adatokra ránézve nyilvánvaló volt. Viszont ez a gyakorlatban nem ennyire nyilvánvaló.

**Ha az lenne a cél, hogy a legkevesebb legyen a távolság a centroidek és a pontok között, minden pont centroid lenne, és 0 lenne az eltérés.**

## Inercia

A centroidek és a centroidekhoz rendelt pontok távolságainak négyzetösszege. Ahogy  $K$  emelkedik, az inercia csökken.



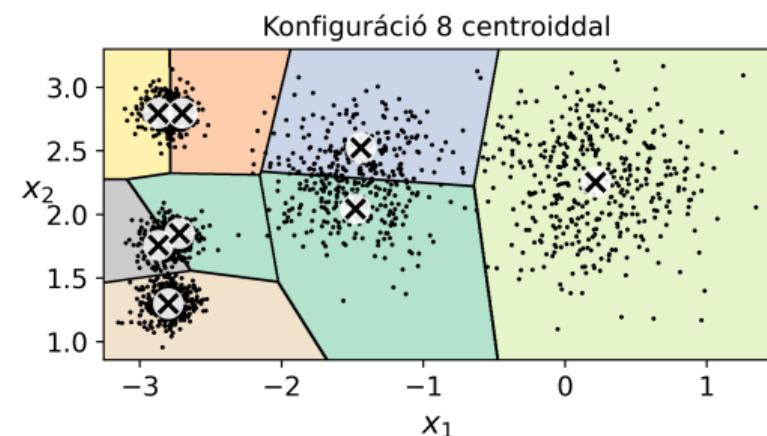
# Optimális klaszterszám megtalálása

A korábbi példában  $K = 5$  volt a klaszterszám, mert ez az adatokra ránézve nyilvánvaló volt. Viszont ez a gyakorlatban nem ennyire nyilvánvaló.

**Ha az lenne a cél, hogy a legkevesebb legyen a távolság a centroidek és a pontok között, minden pont centroid lenne, és 0 lenne az eltérés.**

## Inercia

A centroidek és a centroidekhoz rendelt pontok távolságainak négyzetösszege. Ahogy  $K$  emelkedik, az inercia csökken.



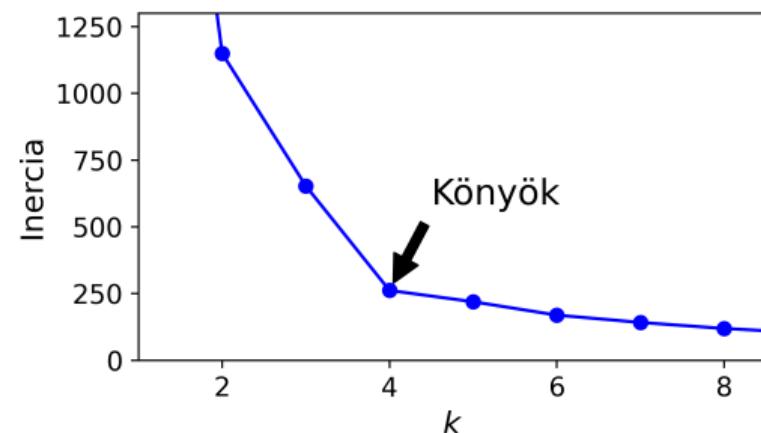
# A könyök módszer

Heurisztikus, vizuális technika a klaszterszám meghatározására.

Az eljárás  $K = 2$  klaszterszámtól indulva különböző  $K$  értékekhez ábrázolja a klaszterkonfiguráció inerciáját.

**A könyökpont ott található, ahol az inercia egy jelentős esése után a csökkenés üteme lelassul.** Az optimális klaszterszám a könyökponthoz tartozó  $K$  érték.

Hátránya, hogy valamikor nincs könyökpont és valamikor több van.



# A sziluett együttható

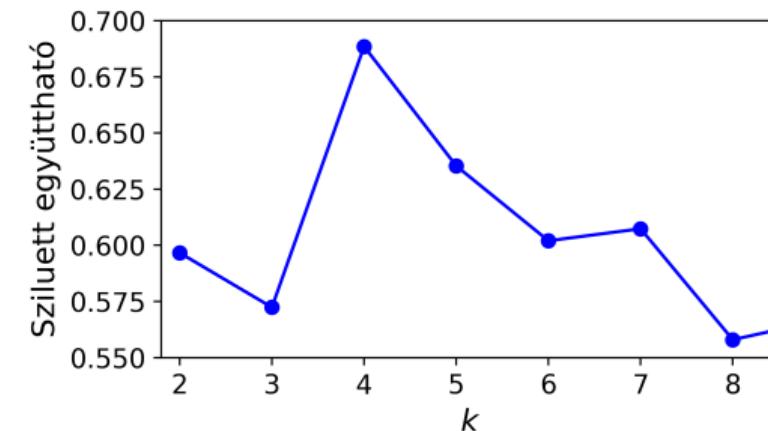
A sziluett alapja a klaszteren belüli szorosság és a többi egyedtől való távolság tartás mértéke.

## Sziluett

Minden  $i \in I$  mintaegyedre a sziluett  $s(i)$ :

$$s(i) = \frac{b(i) - a(i)}{\max(b(i), a(i))}$$

- $a(i)$ : Átlagos távolság  $i$  mintaegyed, és az összes vele egy klaszterben lévő egyed között
- $b(i)$ : Átlagos távolság  $i$  mintaegyed és az összes, egyéb klaszterben lévő mintaegyed között



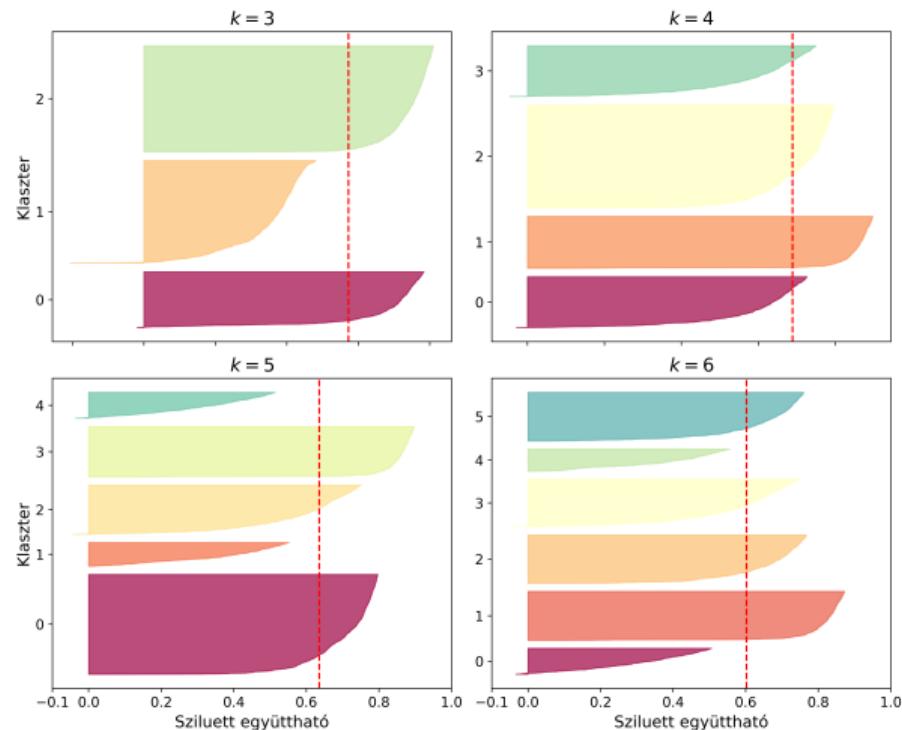
A sziluett tartománya:  $s(i) \in [-1, 1]$ , ahol:

- $-1$ : Teljes félreosztályozás
- $0$ : Közömbös osztályozás
- $1$ : Tökéletes osztályozás

# Teljes konfigurációs tér sziluettje

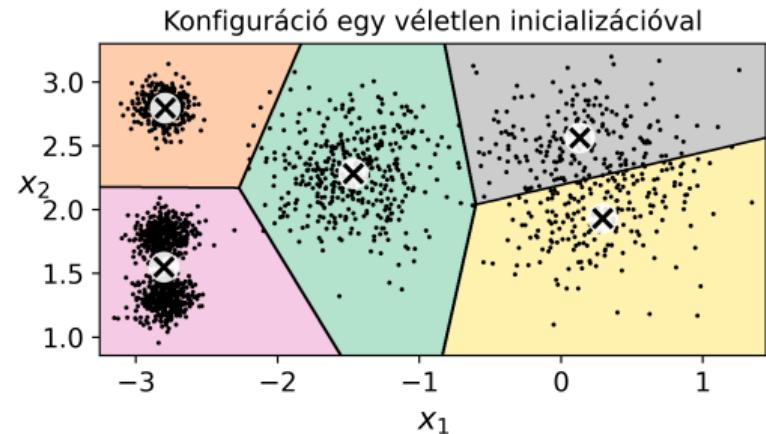
A sziluett nem csak mintaegyedekre értelmezhető, hanem terekre is.

Egy térben helyet foglaló mintaegyedek halmazára vonatkoztatott sziluett-koefficiens a térben megtalálható egyedi pontok sziluettjének átlaga.



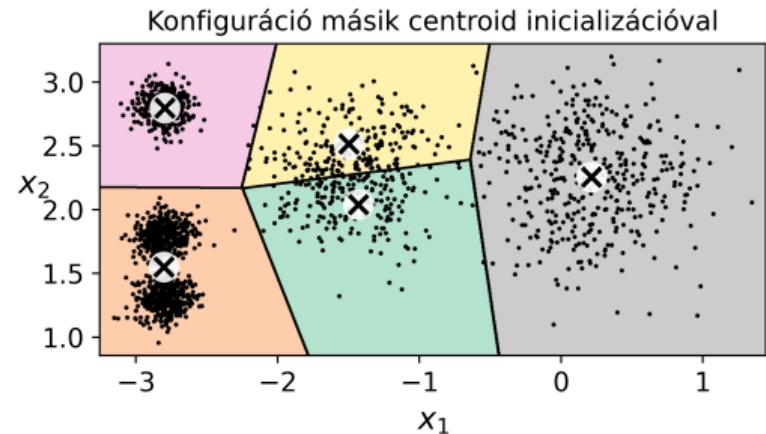
# A $K$ -közép limitációi

Az algoritmust sokszor kell lefuttatni, hogy a szuboptimális konfigurációkat el lehessen kerülni. A klaszterek számát kézzel kell meghatározni.



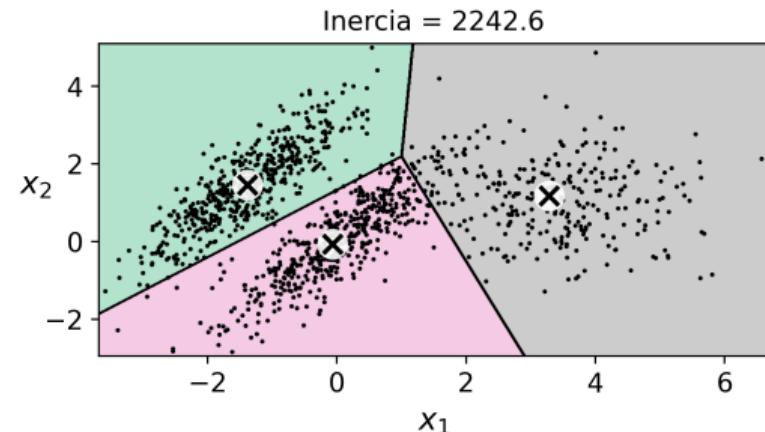
# A $K$ -közép limitációi

Az algoritmust sokszor kell lefuttatni, hogy a szuboptimális konfigurációkat el lehessen kerülni. A klaszterek számát kézzel kell meghatározni.



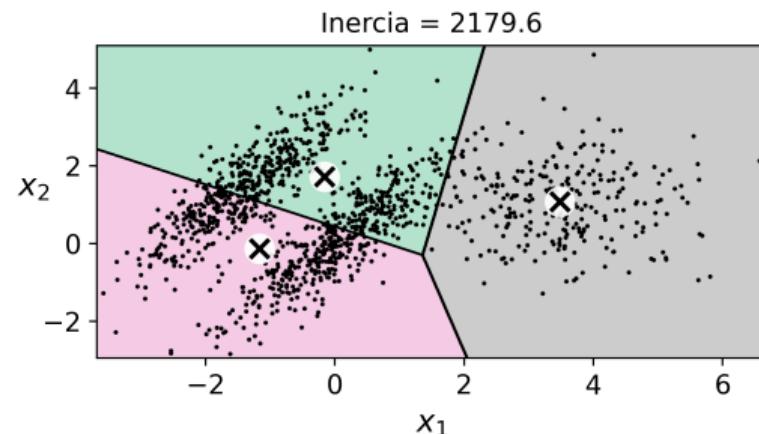
# A $K$ -közép limitációi

A  $K$ -közép továbbá nem viselkedik megfelelően, amikor a klaszterek méretei és alakjai változatosak, eltérő sűrűségűek. Ez abból ered, hogy a modell két pont távolságát veszi alapul.



# A $K$ -közép limitációi

A  $K$ -közép továbbá nem viselkedik megfelelően, amikor a klaszterek méretei és alakjai változatosak, eltérő sűrűségűek. Ez abból ered, hogy a modell két pont távolságát veszi alapul.



Dimenzionalitás  
ooooo

Manifold tanulás  
oooo

Dimenziócsökkentés  
oooooo

Klaszteranalízis  
oooooooo

DBSCAN  
●oo

## 1 Dimenzionalitás

## 2 Manifold tanulás

## 3 Dimenziócsökkentés

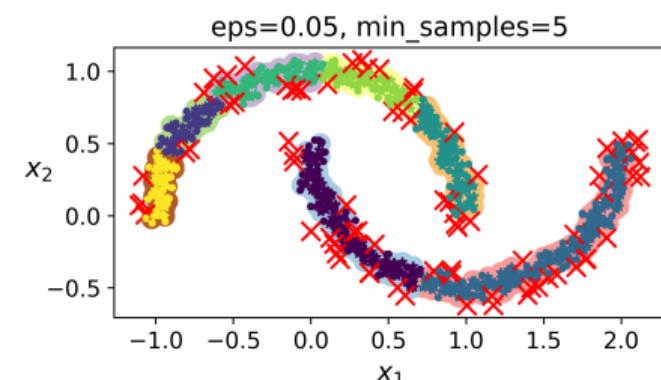
## 4 Klaszteranalízis

## 5 DBSCAN

# DBSCAN

A DBSCAN egy sűrűség alapú klaszterezési algoritmus, amely már irreguláris alakzatú klasztereket is képes megbecsülni:

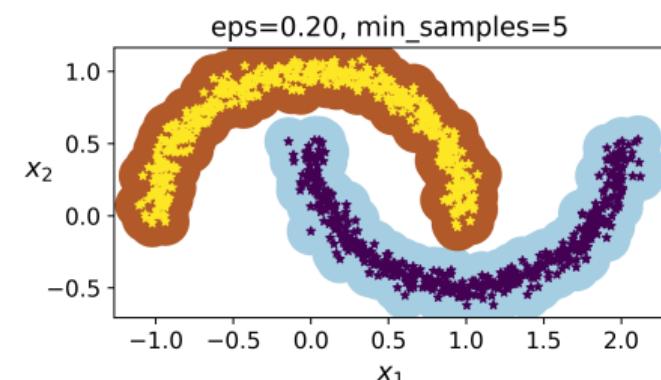
- ① minden egyedhez tartozóan kiszámolja, hány egyed található az  $\varepsilon$  sugarú környezetében ( $\varepsilon$  szomszédság)
- ② Ha `min_samples` küszöbnyi egyed egy  $\varepsilon$  szomszédságban, a pontból mag-egyed válik
- ③ minden mintaegyed, amely egy mag-egyed szomszédságában van, a mag-egyed klaszterébe tartozik
- ④ A szomszédos mag-egyedek és azok szomszédságai egy klaszterbe tartoznak
- ⑤ minden mintaegyed, ami nincs egy szomszédságában sem, anomáliának számít



# DBSCAN

A DBSCAN egy sűrűség alapú klaszterezési algoritmus, amely már irreguláris alakzatú klasztereket is képes megbecsülni:

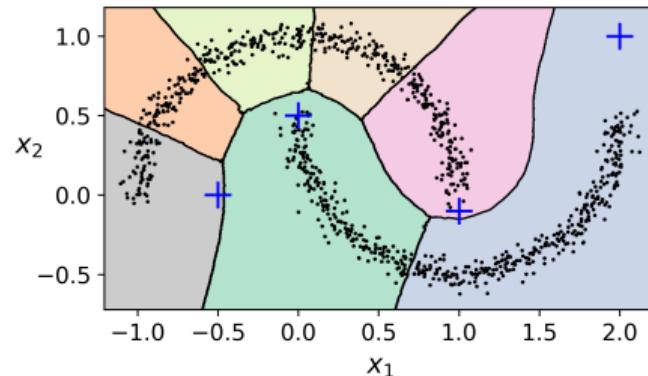
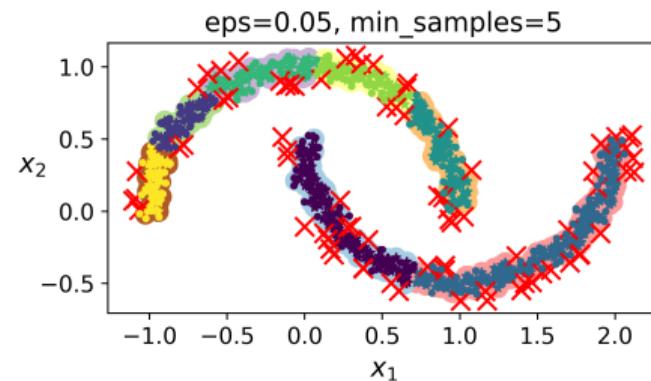
- ① minden egyedhez tartozóan kiszámolja, hány egyed található az  $\varepsilon$  sugarú környezetében ( $\varepsilon$  szomszédság)
- ② Ha `min_samples` küszöbnyi egyed egy  $\varepsilon$  szomszédságban, a pontból mag-egyed válik
- ③ minden mintaegyed, amely egy mag-egyed szomszédságában van, a mag-egyed klaszterébe tartozik
- ④ A szomszédos mag-egyedek és azok szomszédságai egy klaszterbe tartoznak
- ⑤ minden mintaegyed, ami nincs egy szomszédságában sem, anomáliának számít



# Predikció DBSCAN modellel

Meglepő módon a DBSCAN implementációjának nincs `predict()` metódusa, ezért **nem tudja egy új egyedről eldönteni, melyik klaszterhez tartozik.**

Ennek az a mögöttes rationalitása, hogy a DBScan által felismert klasztereken kellően egyszerű futtatni egy osztályozó algoritmust, mint a `KNeighborsClassifier`. Így képes új egyedeket osztályozni.



# Predikció DBSCAN modellel

Meglepő módon a DBSCAN implementációjának nincs `predict()` metódusa, ezért **nem tudja egy új egyedről eldönteni, melyik klaszterhez tartozik.**

Ennek az a mögöttes rationalitása, hogy a DBScan által felismert klasztereken kellően egyszerű futtatni egy osztályozó algoritmust, mint a `KNeighborsClassifier`. Így képes új egyedeket osztályozni.

