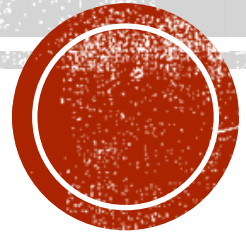
K-MEANS 相關的分群演算法研究

The study of K-means related clustering algorithm



資訊工程學系 大四 B043040003 范真瑋
資訊工程學系 大四 B043040020 張哲魁
資訊工程學系 大四 B043040039 周家池
資訊工程學系 大四 B043040044 吳俊忻

簡介

K-MEANS

優

- 時間複雜度低
- 空間複雜度低
- 對於高效資料集，簡單、高效

缺

- 易陷於 區域最佳
- 需預設 K 值
- 對 初始值 敏感
- 對 noise 及 outlier 敏感
- 只適用 numerical 類型
- 不能解 non-convex 資料

K-MEANS

K-means 的問題	改善演算法
對初始值敏感	K-means ++
對 noise , outlier 敏感	K-medoids K-medians
不可解 non-convex 資料	Kernel K-means

方法

方法

K – means ++

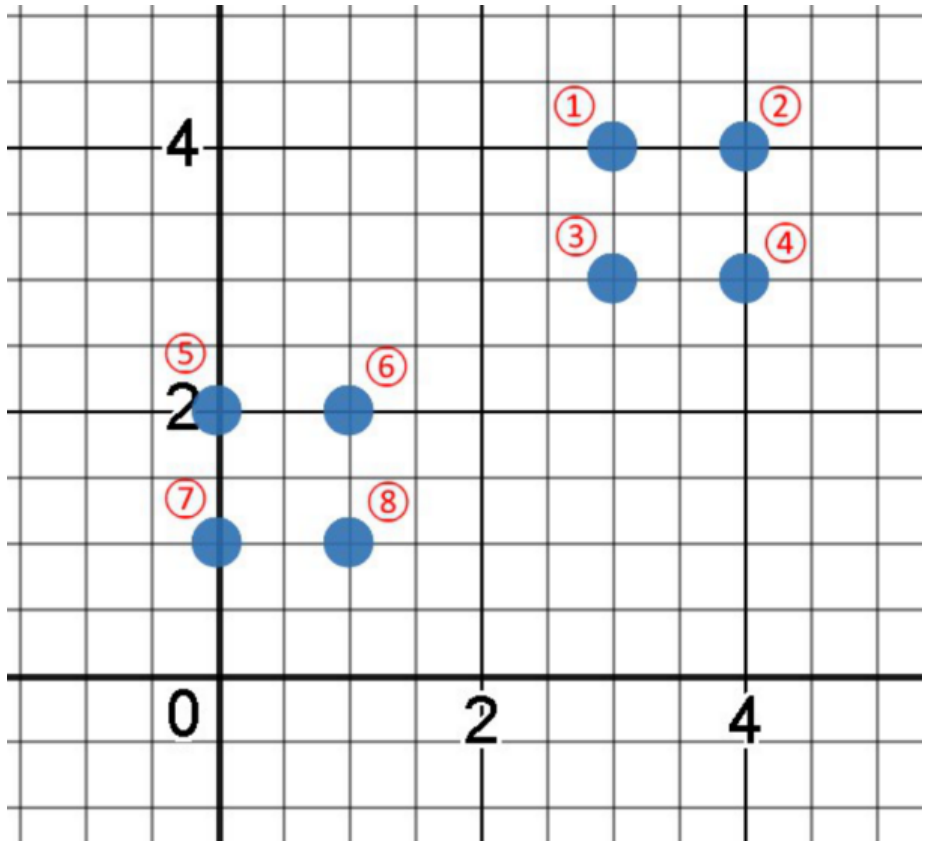
K-means ++

對選取初始中心點的方法做改進，希望 K 個群的中心互相越遠越好。

1. 從資料集中隨機選一個作為初始中心

2. 計算每個樣本與目前已有的中心的最短距離D(x)，再計算每個樣本被選為下一個中心的機率 $\frac{D(x)}{\sum_{x \in X} D(x)}$，最後以該機率隨機選出下一個中心

3. 重複第二步直到共選出 K 個中心



K-means ++

實作:

算出每個樣本被選為下一個中心的機率後，再計算累積機率。接著 隨機產生一個 0~1 之間的數，判斷屬於哪個區間，並選出對應的點作為 下一個中心。

	①	②	③	④	⑤	⑥	⑦	⑧
$\boldsymbol{D(x)}$	$2\sqrt{2}$	$\sqrt{13}$	$\sqrt{5}$	$\sqrt{10}$	1	0	$\sqrt{2}$	1
$\boldsymbol{D(x)^2}$	8	13	5	10	1	0	2	1
$\boldsymbol{P(x)}$	0.2	0.325	0.125	0.25	0.025	0	0.05	0.025
$\boldsymbol{Sum}$	0.2	0.525	0.65	0.9	0.925	0.925	0.975	1

方法

K - medoids

K-medoids

對 update 進行改進，由原本的算平均來更新新的群心，改為新的中心點就是與其他點的距離和最小的那一個，其餘步驟皆與 K-means 相同。

時間複雜度較高，為$O(n^2)$ ，因為 update 時還要去算每個點的距離，不像是 K-means 直接取平均值。

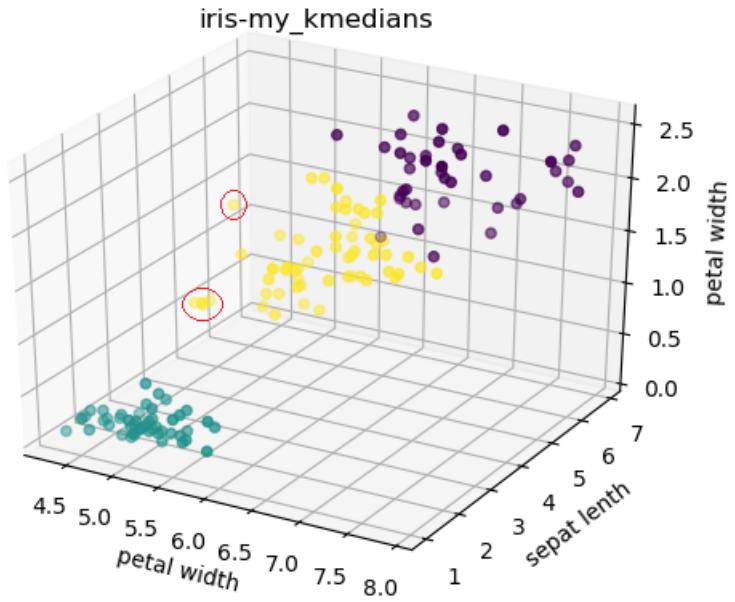
方法

K - medians

K - medians

與 K-means 不同的地方:

- 計算群心是利用中位數而不是平均值，可避免 outlier 造成影響
- 計算距離則是用 Manhattan distance



K - medians

優點:

- 若要計算的距離為 Taxicab metric，則適合用 K-medians
- 計算距離的時間複雜度為 O(KN)

  K 為群心數、N 為資料數

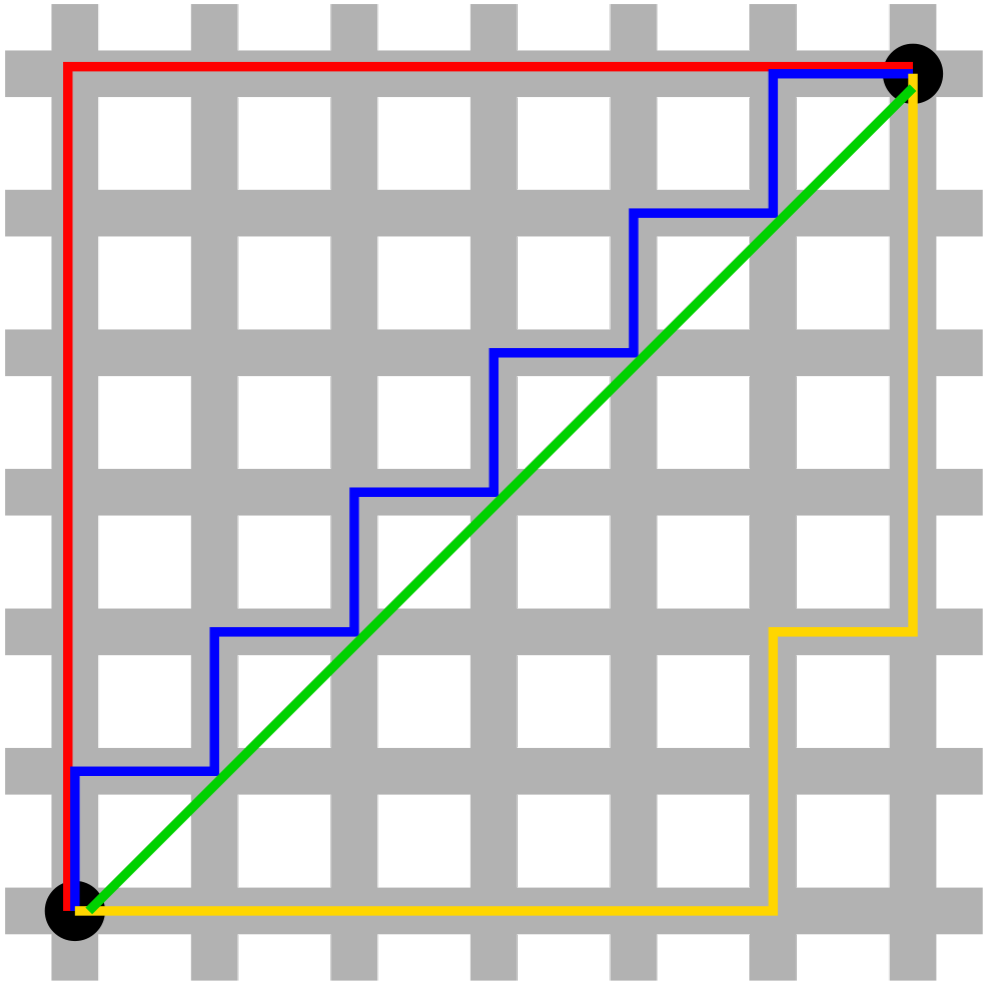


Image from : https://zh-yue.wikipedia.org/wiki/File:Manhattan_distance.svg

方法

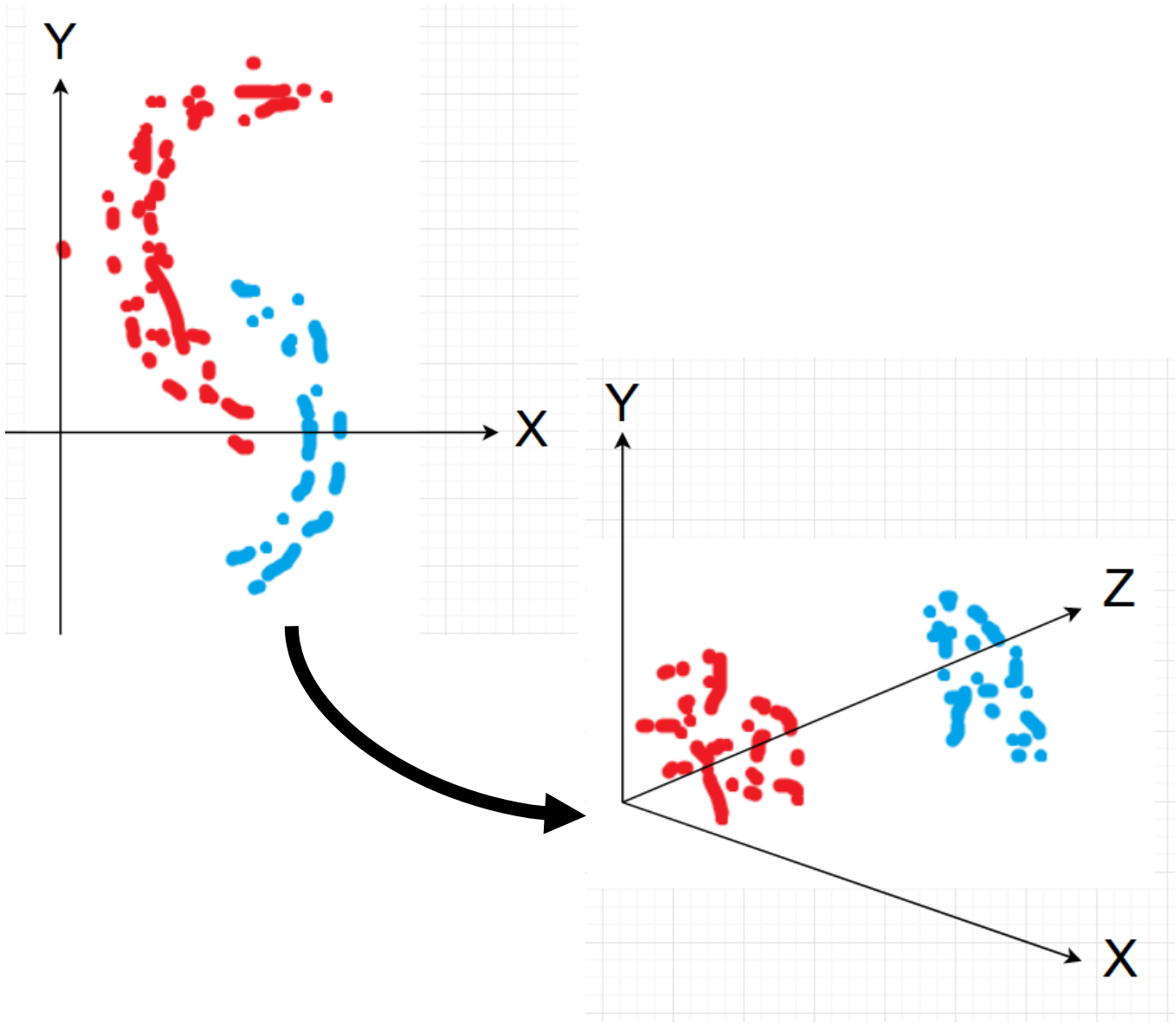
Kernel K - means

Kernel K-means

If we can….

For each points $Xi \in R^n$

, do $Xi \rightarrow \varphi i$ ($\varphi i \in R^f$, $f > n$)



Kernel K-means

In K-means:

target: $\arg\min(\|Xi - Ck\|^2)$

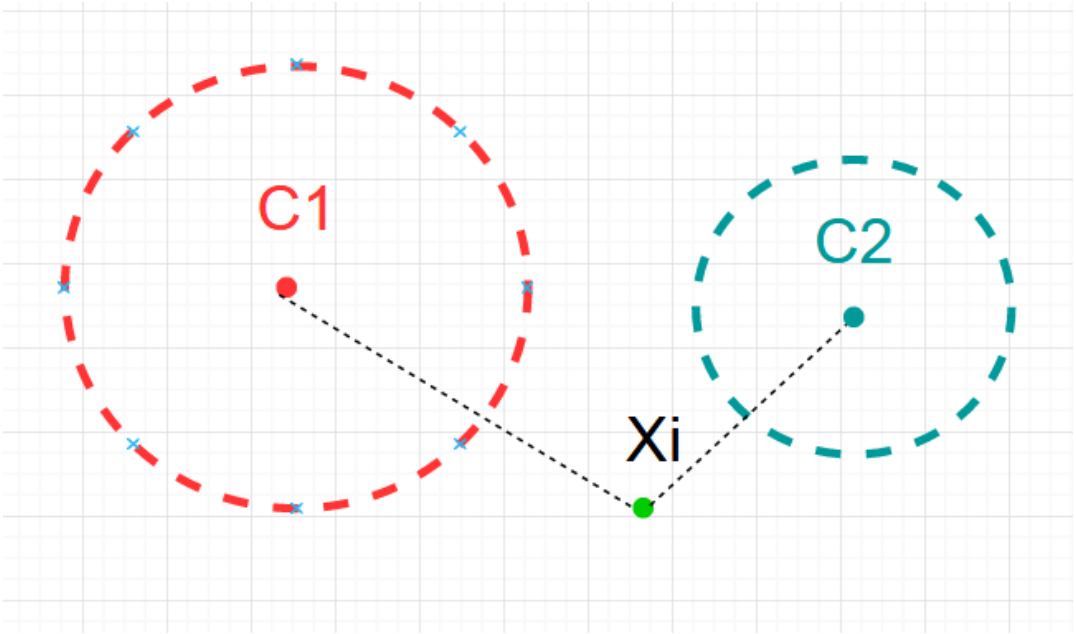
Distance(Xi, Xj) $\in R^1$

$$\|Xi - Xj\|^2 = \|Xi\|^2 - 2 * Xi \cdot Xj + \|Xj\|^2$$

So, we can make a function

$$K(Xi, Xj) = \varphi i \cdot \varphi j$$

just $R^n \to R^1$ not $R^n \to R^f \to R^1$



Kernel K-means

In kernel K-means:

we do not have center Ck ,so

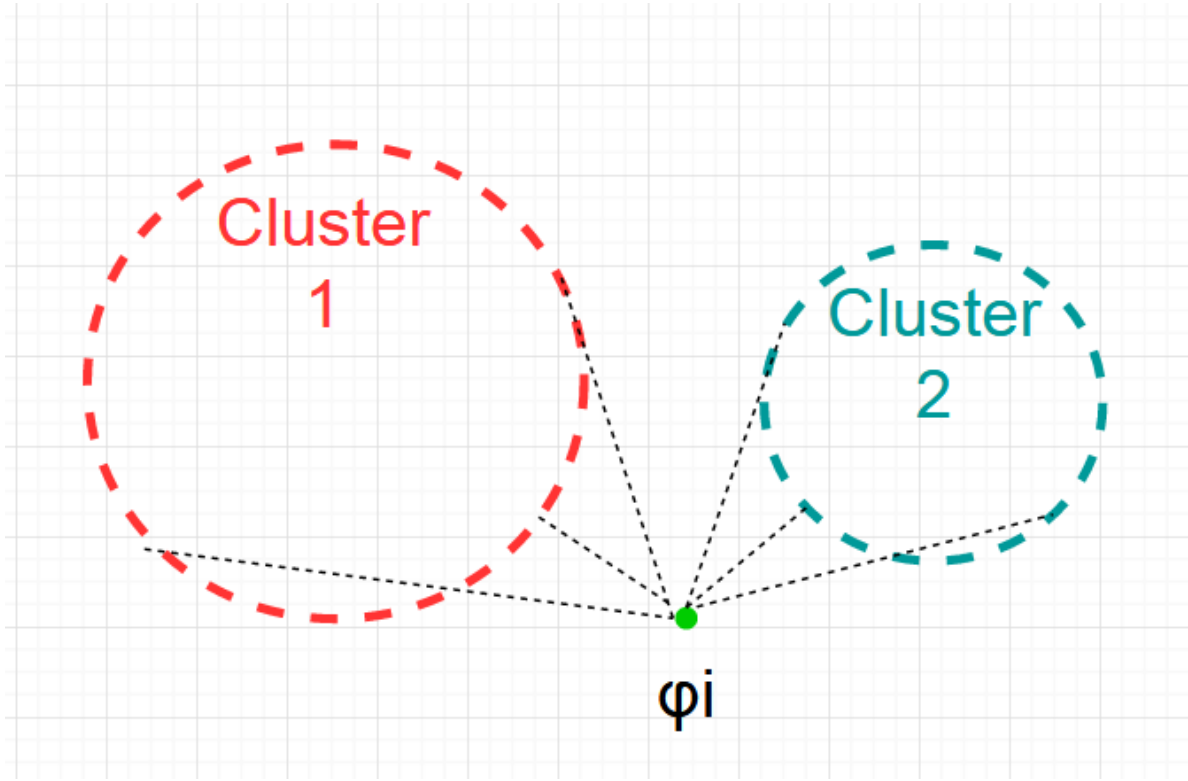
target:

$$\arg\min\left(\frac{\sum(\|\varphi i - \varphi j\|^2)}{num\ of\ cluster\ k}\right),\ \forall \varphi j \in Cluster\ K$$

Complex :

$$O(n^2 * ToK)$$

ToK 表 kernel function 的計算複雜度



Kernel K-means

kernel function:

Gaussian Radio Basis (RBF):

$$K(x,y) = e^{-\frac{\|x-y\|^2}{2\sigma^2}}$$

, σ 是可調參數

Polynomial :

$$K(x,y) = (x \cdot y + c)^d$$

, c與d是可調參數

結果

評分標準

SSE

- Range : 0 ~ ∞
- The smaller, the better

completeness score (sklearn)

- Range : 0 ~ 1
- The larger, the better

- Ex : Ground Truth = [0 , 0 , 1 , 1]

  predict1 = [0 , 1 , 0 , 1]

  --score = 0.0

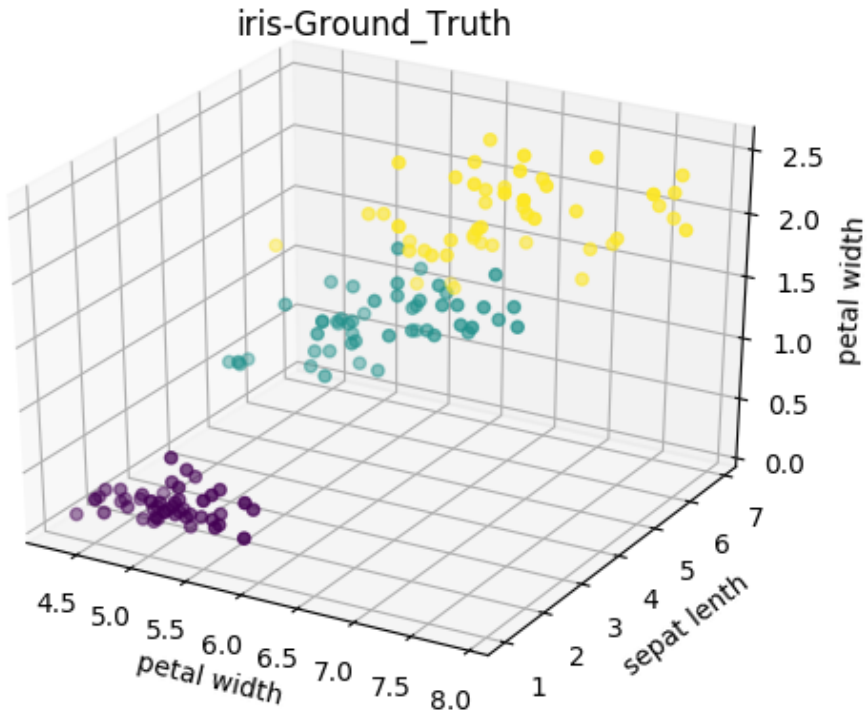
  predict2 = [1 , 1 , 0 , 0]

  --score = 1.0

Iris

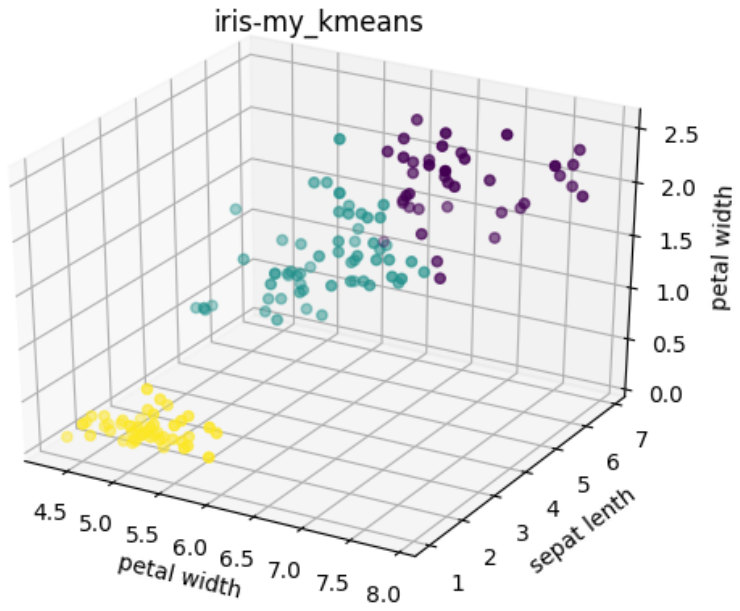
Image from : https://commons.wikimedia.org/wiki/File:Iris_sanguinea.JPG

Iris

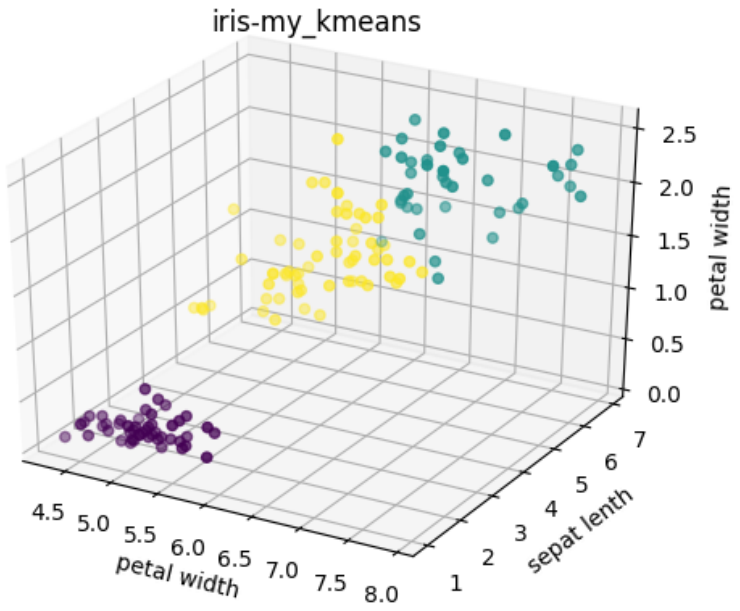

iris-Ground_Truth
petal width
2.5
2.0
1.5
1.0
0.5
0.0
4.5
5.0
5.5
6.0
6.5
7.0
7.5
8.0
petal width
1
2
3
4
5
6
7
sepat lenth

Iris

K-means

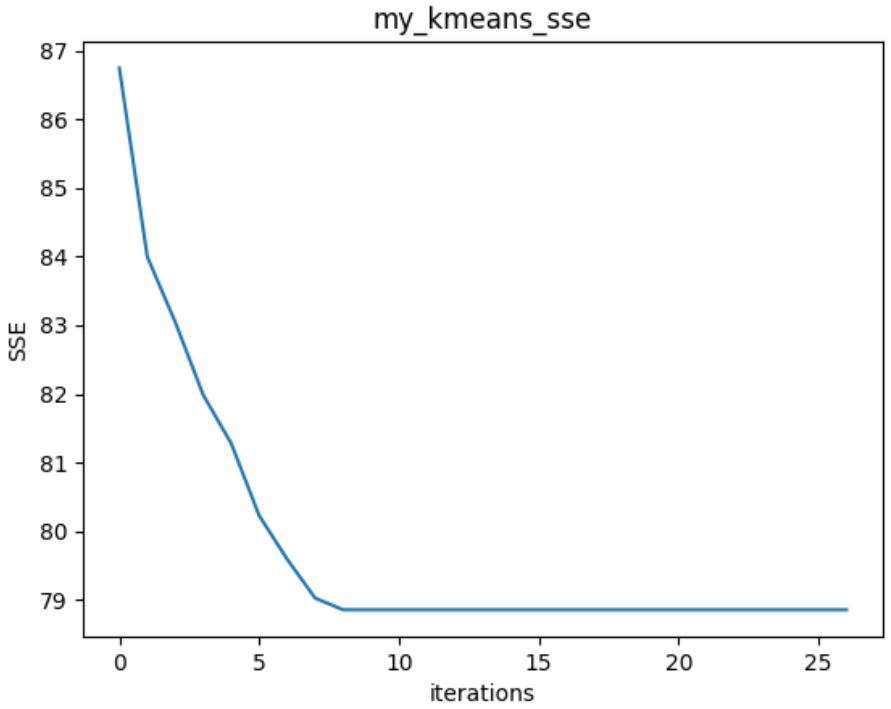


K-means++

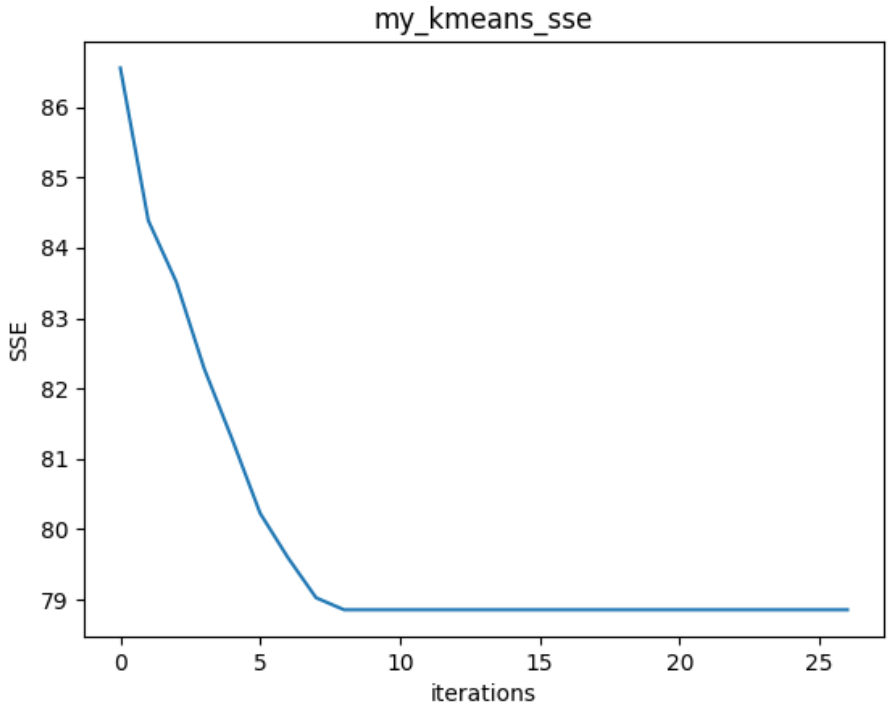


Iris

K-means

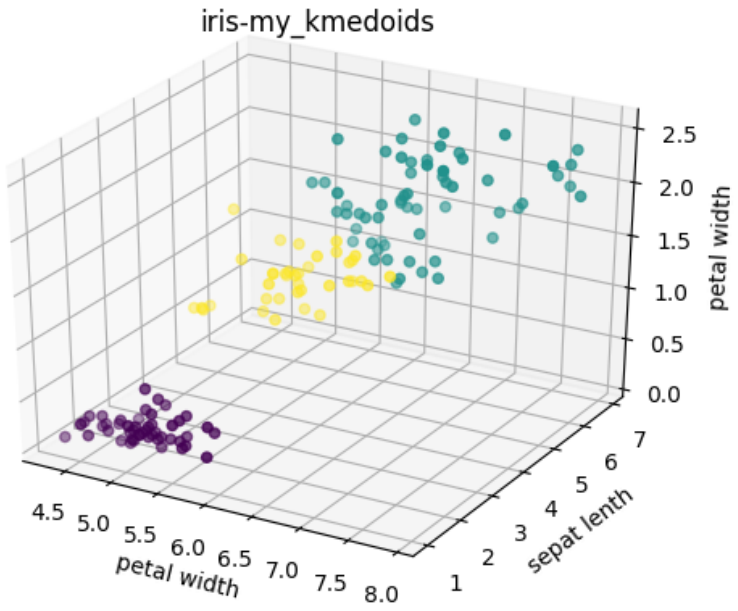


K-means++

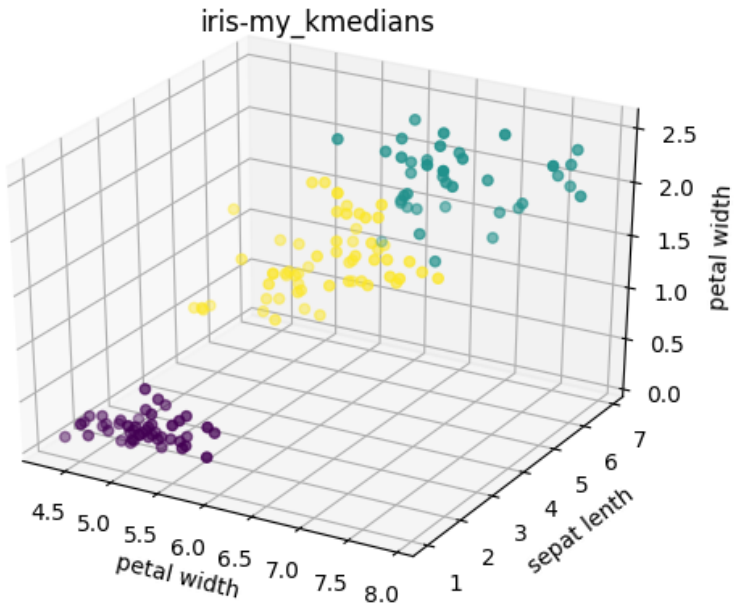


Iris

K-medoids

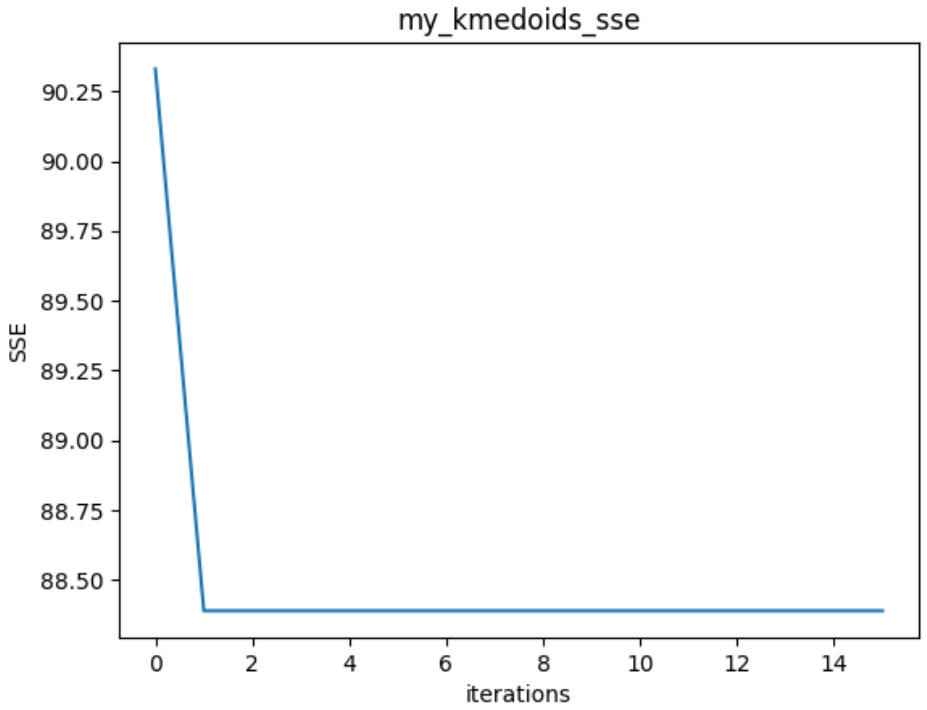


K-medians

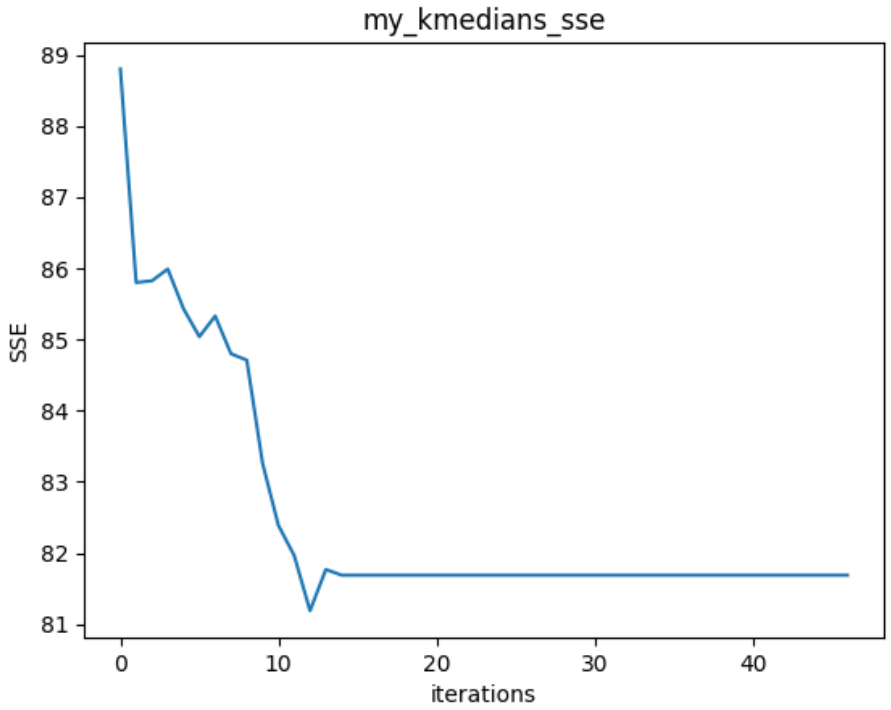


Iris

K-medoids

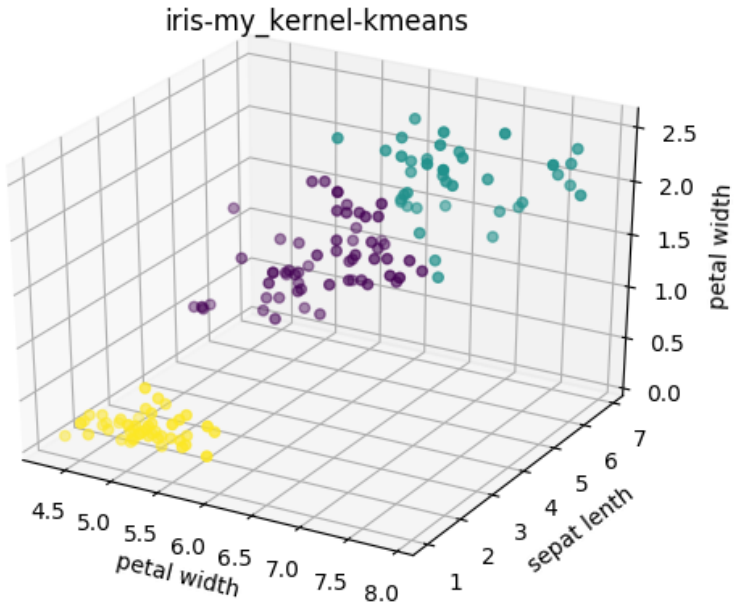


K-medians

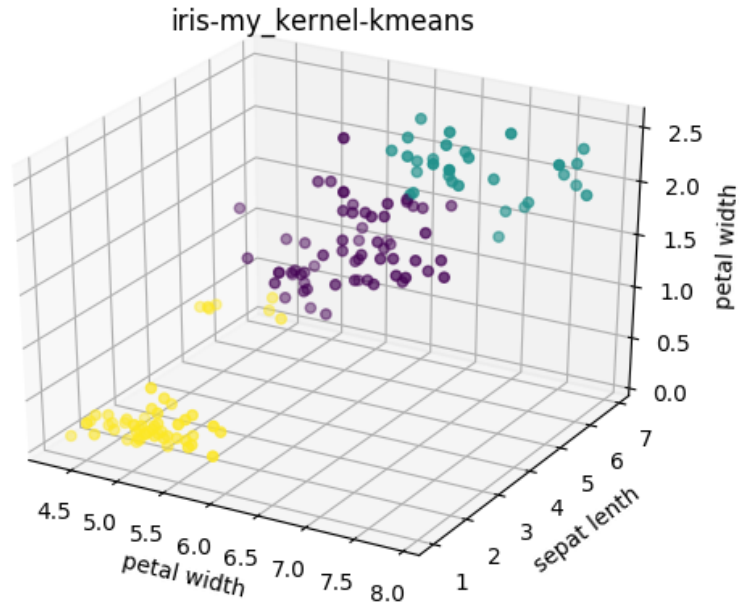


Iris

Kernel K-means (RBF)

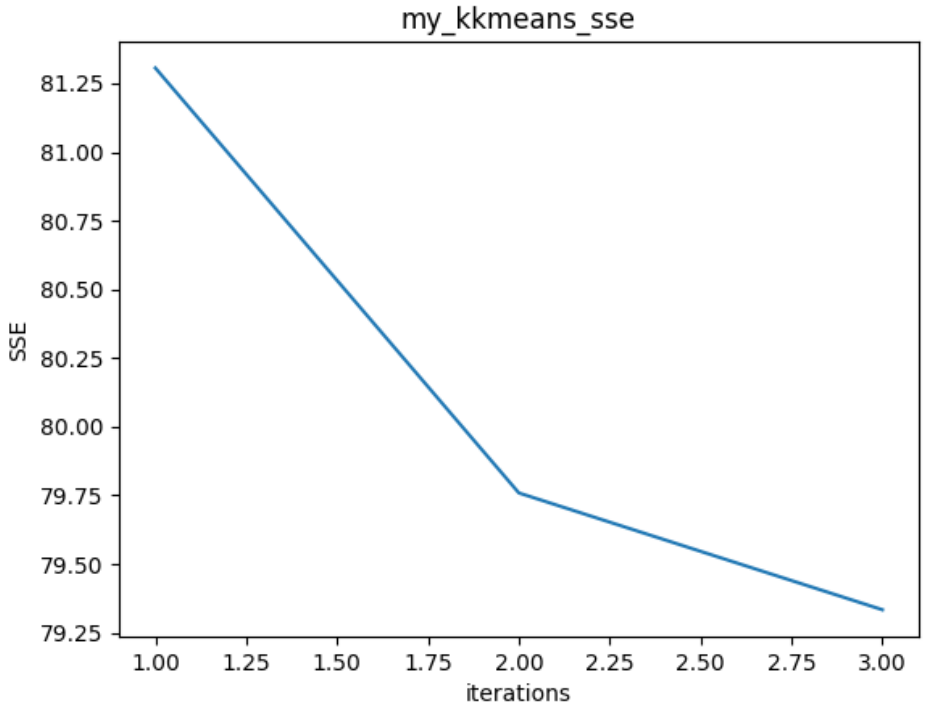


Kernel K-means (Poly)

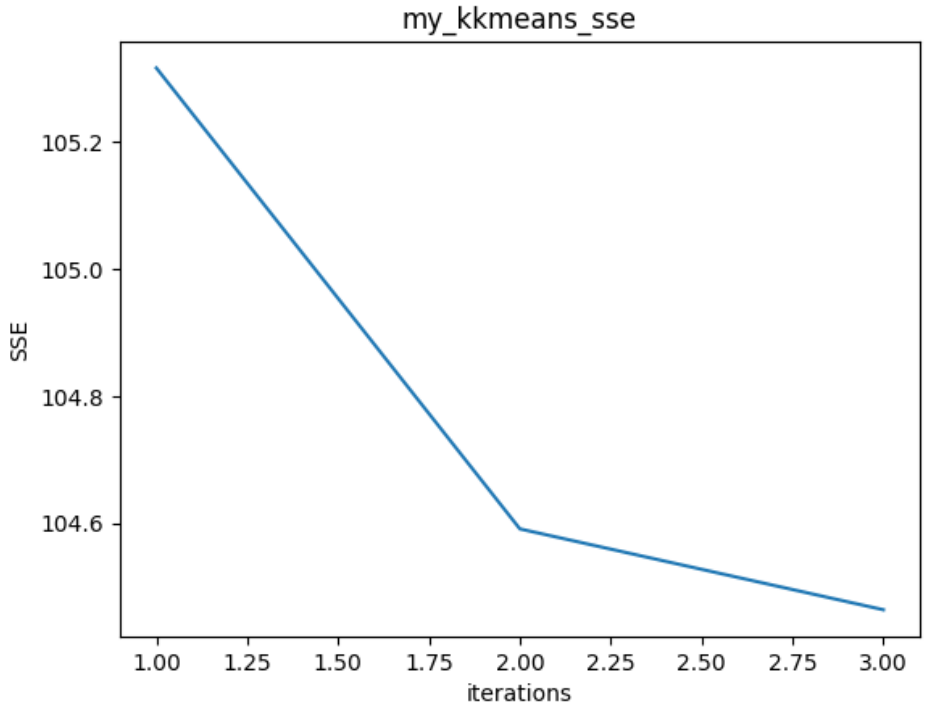


Iris

Kernel K-means (RBF)



Kernel K-means (Polynomial)



Iris

Iris (K = 3)

	K-means	K-medoids	K-medians	KK-means
Random / RBF	78.855666	88.39000	81.69000	78.855664
K-means++ / POLY	78.851441	84.63000	81.69000	97.399080

※KK-means : Kernel K-means (SSE)

※RBF 與 POLY 為 Kernel K-means 的 kernel funciton

Iris

Iris-k3 completeness score (sklearn)

	K-means	**K-medoids**	**K-medians**	**KK-means**
Random / RBF	0.747486	0.792722	0.771791	0.747486
K-means++ / POLY	0.747486	0.792722	0.771791	0.659058

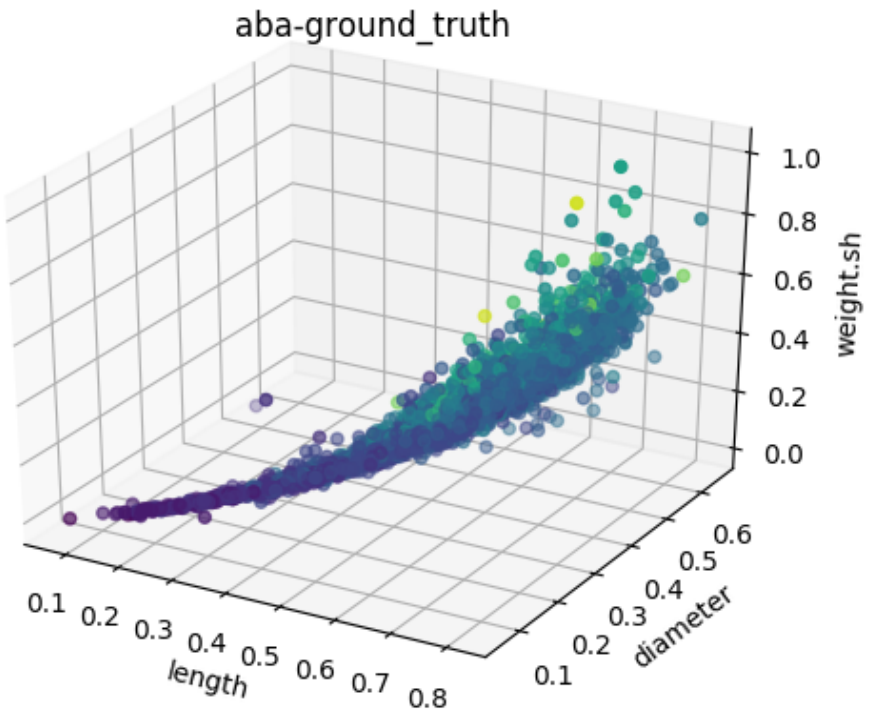
※KK-means : Kernel K-means (score)

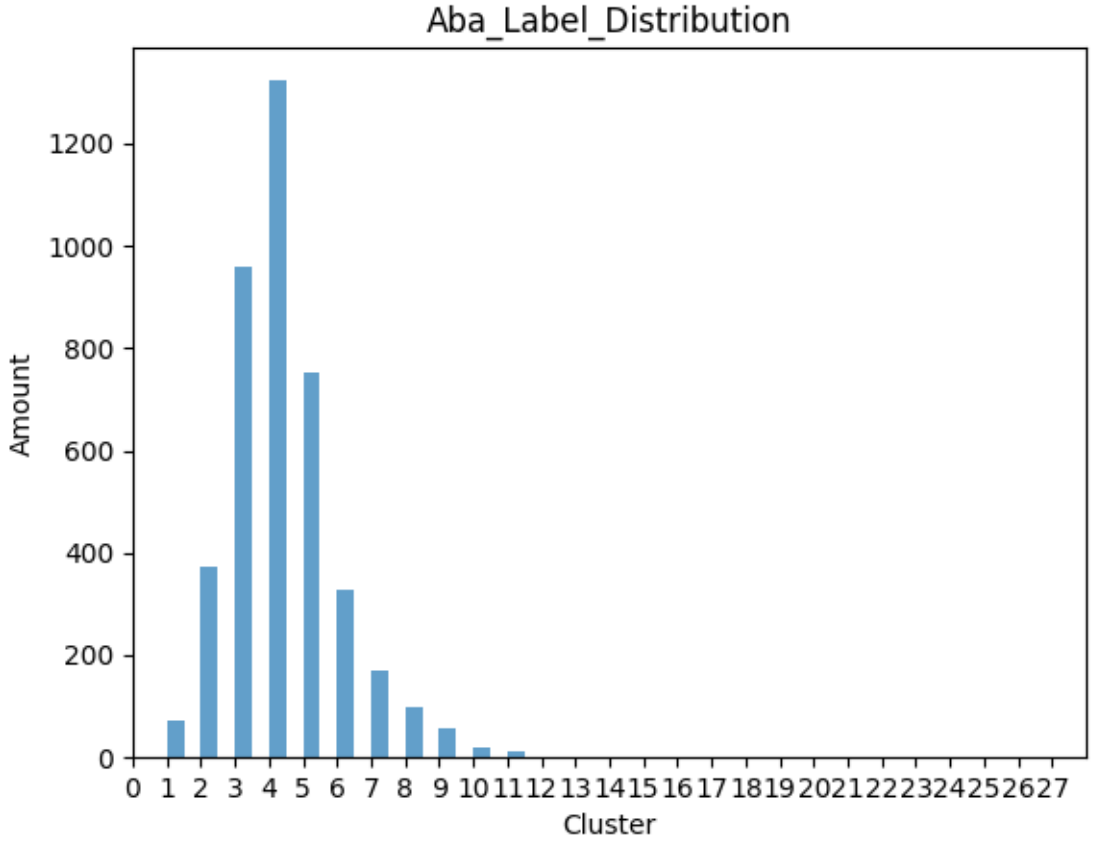
※RBF 與 POLY 為 Kernel K-means 的 kernel funciton

Abalone (K=28)

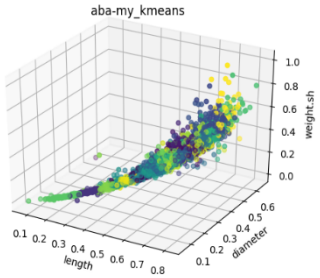
Image from : https://www.maxpixel.net/Abalone-3-Abalone-2-Abalone-2495964

Abalone (K=28)

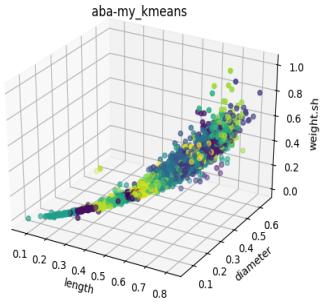

aba-ground_truth
length
diameter
weight.sh
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.0
0.2
0.4
0.6
0.8
1.0


Aba_Label_Distribution
Amount
Cluster
0
200
400
600
800
1000
1200
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27

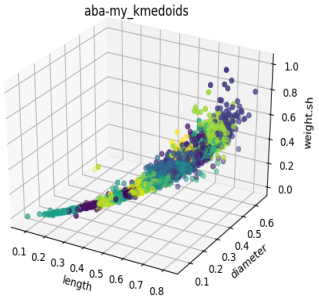
Abalone (K=28)



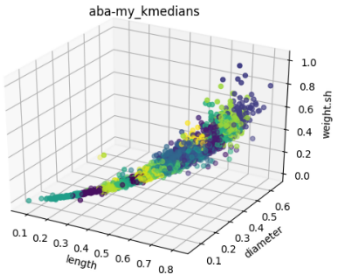
K-means



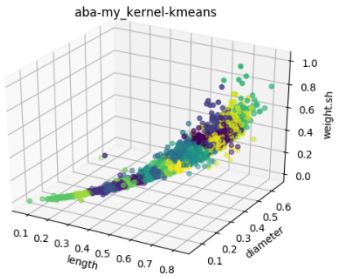
K-means++



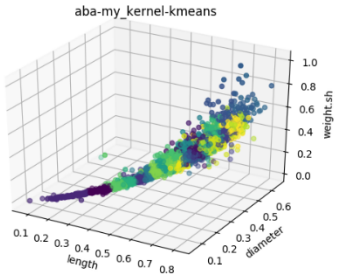
K-medoids



K-medians

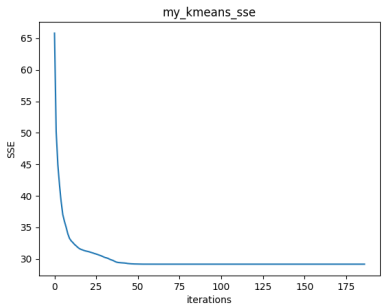


KK-means(RBF)

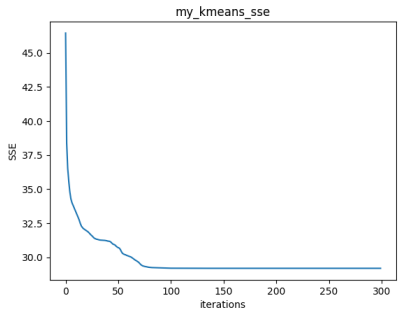


KK-means(poly)

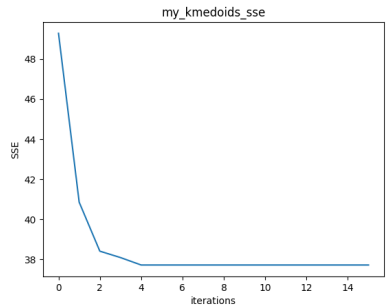
Abalone (K=28)



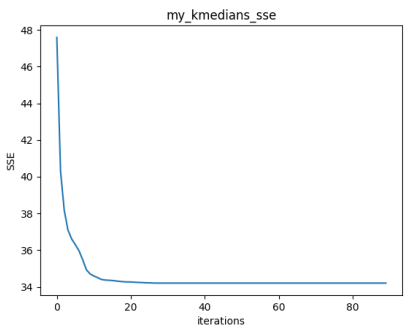
K-means



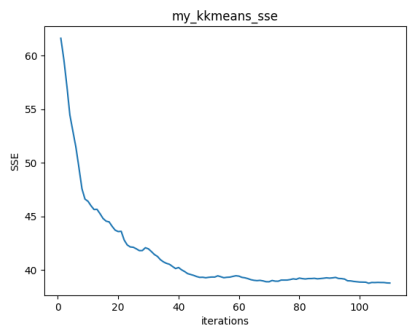
K-means++



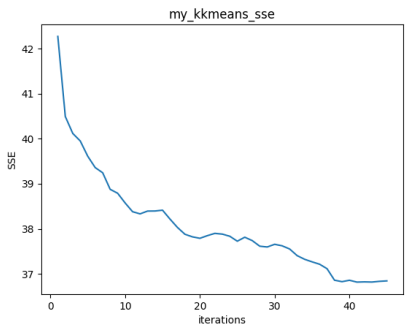
K-medoids



K-medians



KK-means(RBF)



KK-means(poly)

Abalone (K=28)

Abalone (K = 28)

	K-means	K-medoids	K-medians	KK-means
Random / RBF	28.146570	38.776020	32.525794	29.859349
K-means++ / POLY	27.333311	33.747686	28.668192	36.843252

※KK-means : Kernel K-means (SSE)

※RBF 與 POLY 為 Kernel K-means 的 kernel funciton

Abalone (K=28)

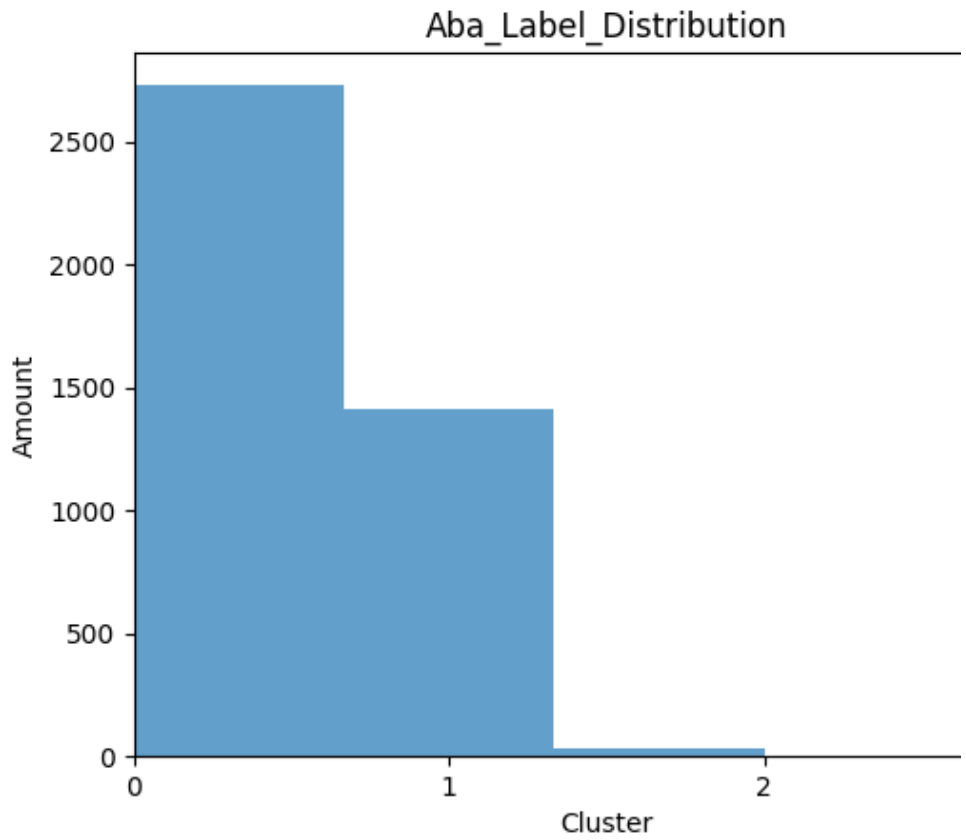
Abalone-k28 completeness score (sklearn)

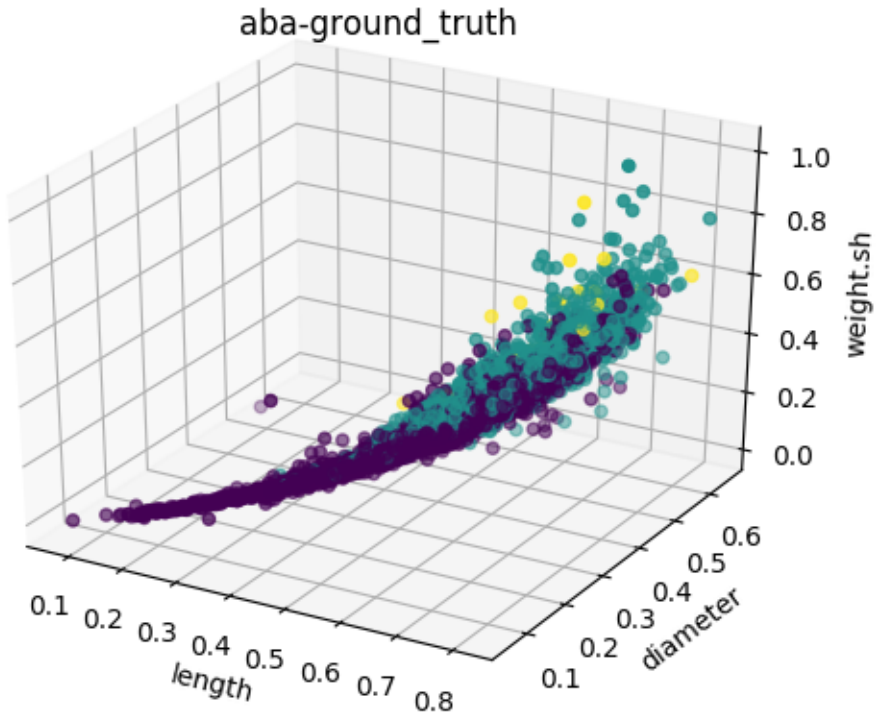
	K-means	K-medoids	K-medians	KK-means
Random / RBF	0.137067	0.130758	0.128056	0.134405
K-means++ / POLY	0.130756	0.126472	0.127896	0.135288

※KK-means : Kernel K-means (score)

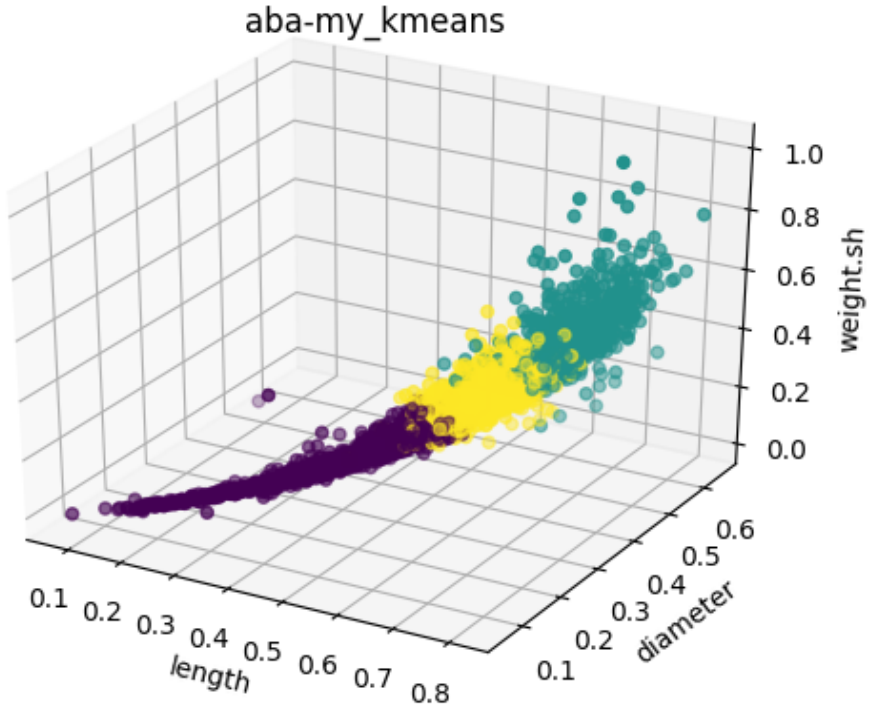
※RBF 與 POLY 為 Kernel K-means 的 kernel funciton

Abalone (K=3)


Aba_Label_Distribution
Amount
2500
2000
1500
1000
500
0
0
1
2
Cluster


aba-ground_truth
weight.sh
1.0
0.8
0.6
0.4
0.2
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
length
0.1
0.2
0.3
0.4
0.5
0.6
diameter

Abalone (K=3)


aba-my_kmeans
1.0
0.8
0.6
0.4
0.2
0.0
weight.sh
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
length
0.1
0.2
0.3
0.4
0.5
0.6
diameter

Abalone (K=3)

Abalone-k3 completeness score (sklearn)

	K-means	K-medoids	K-medians	KK-means
Random / RBF	0.080617	0.079783	0.078580	0.082075
K-means++ / POLY	0.080617	0.078537	0.077928	0.078428

※KK-means : Kernel K-means (score)

※RBF 與 POLY 為 Kernel K-means 的 kernel funciton

Time Cost

Image from : https://pxhere.com/en/photo/835791

Time Cost

Abalone-k3 Time Cost

	K-means	K-medoids	K-medians	KK-means
Random / RBF	13.6858	16.9341	15.1587	314.1200
K-means++ / POLY	9.2037	7.2611	6.7032	329.8859

※KK-means : Kernel K-means (second)

※RBF 與 POLY 為 Kernel K-means 的 kernel funciton

參考資料

參考資料

Arthur, D. & VassilvitsKii, S. (2007). K-means++: The Advantages of Careful Seeding.

Kaufman, L. & Rousseeuw, P.J. (1987). Clustering by means of Medoids.

Jain, A. K., & Dubes, R. C. (1988). Algorithms for clustering data.

Englewood Cliffs, NJ: Prentice-Hall.

Ardian Umam. (2017, December 15). Machine-Learning-From-The-Scratch. Retrieved from https://github.com/ardianumam/MachineLearning-From-The-Scratch.

Welling, M. (2013). Kernel K-means and Spectral Clustering.

Wolper, D. H. & Macrea, W. G. (1996). No Free Lunch Theorems for Optimization.

MayuKh, H. (2010). Age of Abalones using Physical Characteristics:A Classification Problem.

廖韋雅、詹祥麟、薛友仁(2007)。集群方法與集群指標關係之研究。

感謝聆聽