سومین کنفرانس ملّی
محاسبات توزیعی و پردازش داده های بزرگ
3rd National Conference on Distributed Computing and Big Data Processing

ارائه یک روش بهبود یافته Kernel K means با روش Min-Max

امین گلزاری اسکوئی[۱]، عسگرعلی بویر[۲]، هیوا ابراهیم‌زاده[۳]

[۱] دانشجوی کارشناسی ارشد، دانشکده فناوری اطلاعات و مهندسی کامپیوتر دانشگاه شهید مدنی آذربایجان – تبریز – ایران
a.golzari@azaruniv.ac.ir

[۲] استادیار، دانشکده فناوری اطلاعات و مهندسی کامپیوتر دانشگاه شهید مدنی آذربایجان – تبریز – ایران
a.bouyer@azaruniv.ac.ir

[۳] دانشجوی کارشناسی ارشد، دانشکده فناوری اطلاعات و مهندسی کامپیوتر دانشگاه شهید مدنی آذربایجان – تبریز – ایران
h.ebrahimzade@azaruniv.ac.ir

چکیده

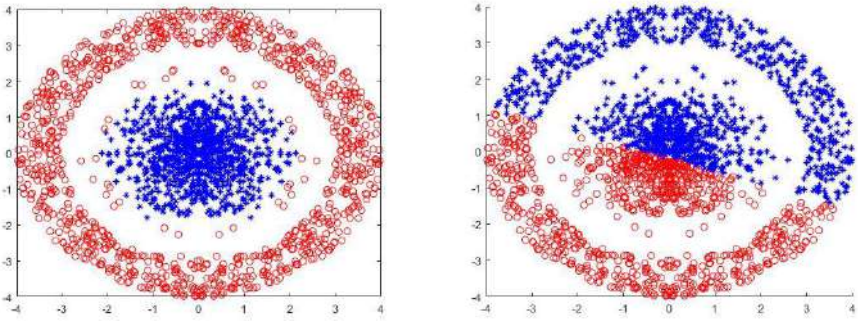
مینیمم کردن مجموعه واریانس داخل خوشه‌ای، یکی از محبوب‌ترین روش‌های خوشه بندی است. از این رو به دلیل انتخاب نادرست نقاط مرکزی، بهینه سازی محلی ضعیف و ... می‌تواند از محبوبیت آن بکاهد. این روش بهبودی برای روش K means است که در آن سعی شده است مشکل انتخاب نقاط مرکزی با استفاده از روش Min-Max Kernel K means حل شود. الگوریتم ارئه شده، برای متعادل کردن واریانس بین خوشه‌ها وزنی را به خوشه‌ها با توجه به واریانس آنها اختصاص می‌دهد. این الگوریتم روش جدیدی برای وزن دهی خوشه‌ها ارائه می‌کند. وزن‌ها در هر تکرار با توجه به تخصیصاتی که انجام شده است، آپدیت می‌شوند. هدف از وزن دهی به خوشه‌ها جلوگیری از به وجود آمدن خوشه‌هایی با واریانس بالاست. در الگوریتم ارائه شده از الگوریتم Kernel K means استفاده شده است. دلیل این کار، مناسب بودن این روش برای خوشه بندی غیر خطی است. آزمایشات گسترده بر روی چندین دیتاست و مقایسه روش ارئه شده با روش‌های دیگر نشان می‌دهد که الگوریتم ارائه شده به انتخاب نقاط اولیه مناسب حساس نیست و حتی با انتخاب نقاط اولیه نامناسب نیز کارایی بهتری نسبت به دیگر روش‌های مقایسه شده دارد.

کلمات کلیدی

خوشه بندی- خوشه‌های متعادل– K means – Kernel K means

۱– مقدمه

خوشه بندی در بسیاری از رشته‌ها می‌تواند به کار برده شود از جمله بازشناخت الگو، یادگیری ماشین، بیوانفرماتیک و پردازش تصویر[1, 2]. هدف از خوشه بندی داده، به حداقل رساندن فاصله بین نقاط در درون خوشه و به حداکثر رساندن فاصله آنها از خوشه‌های دیگر است. داده‌های انتخاب شده برای خوشه بندی انواع مختلف می‌تواند داشته باشد مثل همگرا[۱]، واگرا[۲]، جداپذیر خطی[۳] و جداپذیر غیرخطی[۴]. بخاطر سادگی و کارایی بالا الگوریتم K means، به یکی از محبوب‌ترین الگوریتم‌ها در میان سایر الگوریتم‌های خوشه بندی تبدیل شده است. الگوریتم K means برای داده‌های همگرا و جداپذیرخطی بیشترین کارایی را دارد ولی برای داده‌های غیرخطی و واگرا نامناسب است. علاوه بر این مشکل، K means مشکل اساسی که دارد، مربوط به انتخاب نقاط اولیه است. راه حل آن بستگی به موقعیت اولیه مرکز خوشه‌ها است، بنابراین بعد از انتخاب نقاط اولیه نامناسب، مینیم سازی محلی ضعیف حاصل می‌شود. برای حل مشکل اول الگوریتم K means، یعنی نامناسب بودن الگوریتم برای داده‌های جداپذیر غیرخطی و واگرا، از الگوریتم Kernel K means استفاده شده است. این الگوریتم قادر به تشخیص خوشه‌هایی است که به صورت غیرخطی از هم جدا شده‌اند و یا واگرا هستند. این الگوریتم برای داده‌های جداپذیر غیرخطی و واگرا بیشترین کارایی را دارد[3, 4] و بهترین خوشه بندی را انجام می‌دهد. جزئیات این روش در بخش ۲-۲ به طور کامل توضیح داده شده است. شکل (۱) مقایسه‌ی الگوریتم Kernel K means با K means را نشان می‌دهد. برای حل مشکل دوم الگوریتم K means، یعنی مشکل مربوط به انتخاب نقاط اولیه، الگوریتم ارائه شده روشی برای وزن دهی خوشه‌ها ارائه می‌کند. وزن‌ها در هر تکرار با توجه به تخصیصاتی که انجام شده است، آپدیت می‌شوند. هدف از وزن دهی به خوشه‌ها جلوگیری از به وجود آمدن خوشه‌هایی با واریانس بالاست. الگوریتم ارائه شده سعی می‌کند در طی تکرار الگوریتم بهترین نقاط را به عنوان مرکز خوشه‌ها، با توجه به وزن اختصاص داده شده به آنها انتخاب کند. این الگوریتم نسبت به نقاط اولیه حساس نبوده و بهترین کارایی را دارد.



شکل (۱) با انتخاب نقاط مرکزی یکسان برای الگوریتم K means و Kernel K means، روش K means (شکل سمت راست) منجر به خوشه بندی ضعیف می‌شود این در حالی است که الگوریتم Kernel K means (شکل سمت چپ) خوشه بندی خوبی را انجام می‌دهد. شکل‌ها و رنگ‌های مختلف نشان دهنده تخصیصات به خوشه‌ها هستند.

۲- کارهای مرتبط

متدهای زیادی برای غلبه کردن به مشکل نقاط اولیه به وجود آمده‌اند. گروه اول به طور سیستماتیک سعی می‌کنند از خوشه بندی ضعیف در طی تکرارها جلوگیری کنند. در مقاله [5] مراکز اولیه به صورت اتفاقی با یک فرایند خاص از میان کل داده‌ها انتخاب می‌شوند. به صورت تئرویک این متد توانایی بهینه کردن خوشه بندی داده‌ها را تا حدودی تامین می‌کند. در بعضی از رویکردها مرکزها به صورت تصادفی انتخاب می‌شوند و خوشه‌ها با توجه به کیفیت ارائه آنها تنبیه می‌شوند این روش در [6, 7] ارائه شده است. خوشه‌هایی که از کیفیت پایینی برخوردارند، در گام‌های بعدی نقاطی جدید که از یک نظم خاص برخوردار هستند به آنها اختصاص می‌یابد. مراکز اولیه‌ای که در جای نامناسب بودند، تغییر یافته و بهترین مراکز انتخاب می‌شوند. نقاط جدید می‌توانند از بوجود آمدن خوشه‌های پرت جلوگیری کنند و باعث تعادل در اندازه خوشه‌ها شوند. راه کارهای شبیه به این الگوریتم را می‌توان در [8, 9] یافت.

گروه دوم از روش‌ها سعی می‌کنند وابستگی به نقاط اولیه را از بین ببرند از این رو تکرارها تا جایی که لازم است ادامه می‌یابد. Global K means [10] و نسخه‌های بعدی آنها [11, 12] روش‌های افزایشی هستند که از یک خوشه تنها شروع کرده و در هر مرحله یک خوشه جدید، به راه حل آنها با توجه به ملاک مناسبی اضافه می‌شود. نسخه‌ای بر پایه Kernel، برای Global K means در [13, 14] موجود است.

روش ارائه شده در این مقاله، در دسته دوم قرار می‌گیرد یعنی الگوریتم سعی می‌کند بهترین نقاط مرکزی را در طی تکرار بدست آورد. این روش نسخه‌ی گسترش یافته ای از مقاله [15] است که در این روش به جای استفاده از K means از Kernel K means استفاده شده است.

۲-۱- الگوریتم K means

برای خوشه بندی دیتاست χ ، به M خوشه جدا از هم، K means سعی می‌کند مجموع واریانس داخل خوشه‌ای را مینیمم کند (۷).

فرض کنیم واریانس به صورت $V_k = \sum_{i=1}^{N} \delta_{ik} \|x_i - m_k\|^2$ و مرکز k امین خوشه به صورت $m_k = \sum_{i=1}^{N} \delta_{ik} x_i / \sum_{i=1}^{N} \delta_{ik}$ تعریف شود که در آن N برابر تعدا نقاط و δ_{ik} نشان می‌دهد که اگر نقطه ی i در خوشه k باشد مقدار آن برابر ۱ در غیر این صورت مقدار آن برابر ۰ خواهد بود:

$$\varepsilon_{sum} = \sum_{K=1}^{M} V_K = \sum_{K=1}^{M} \sum_{I=1}^{N} \delta_{ik} \|x_i - m_k\|^2 \qquad (۱)$$

فرایند این روش به این صورت است که یک نقطه به نزدیکترین نقطه مرکزی اختصاص می‌یابد و در آن خوشه قرار می‌گیرد. سپس این نقاط مرکزی دوباره محاسبه شده و در مراحل بعدی نقاط جدیدی به آنها اختصاص می‌یابند. بر خلاف سادگی این روش مشکل بزرگی که دارد وابستگی این روش به انتخاب نقاط اولیه مرکزی مناسب است[18, 19]، یعنی اگر این انتخاب به درستی انجام نشود باعث می‌شود خوشه بندی از کیفیت مطلوبی برخوردار نباشد و همچنین نقاط اولیه نامناسب منجر به مینیمم محلی ضعیف می‌شود. و معمولا راه حل برای این مشکل چندین بار اجرا الگوریتم است که اگر تعداد خوشه‌ها زیاد باشد یا دیتاست دارای ابعاد زیاد باشد الگوریتم K means با مشکل مواجه خواهد بود.

۲-۲- الگوریتم Kernel K means

روش‌هایی بر پایه Kernel نظیر روش Kernel K means، در تشخیص خوشه‌های واگرا و جداپذیر غیرخطی همواره بهترین کارایی را داشته اند. در واقع این روش یک بهبودی غیر خطی برای روش K means است[16]. برای اولین بار در [4] این روش ارائه شد. در گام اول، نقاط داده از فضای ورودی[۵] به یک فضای ویژگی[۶] با ابعاد بالا از طریق تبدیل غیرخطی $\emptyset$[۷] نگاشت می‌شوند و سپس الگوریتم سعی می‌کند خطای خوشه بندی در فضای ویژگی را مینیمم کند. فاصله بین نقاط داده و مرکز خوشه‌ها در فضای ویژگی می‌تواند با استفاده از تابع Kernel بدون در نظر گرفتن فرم صریحی از تبدیل، محاسبه شود[17]. این بخاطر این است که ضرب نقطه‌ای[۸] بین دو نقطه X و Y در فضای ویژگی

$\emptyset(x).\emptyset(y)$، می‌تواند با تابع Kernel یعنی $K(X,Y)$، محاسبه شود به طوری که $K:D\times D\rightarrow\mathbb{R}$ است. تعدادی Kernel های استاندارد به صورت زیر وجود دارد.

- Polynomial kernel of degree p: $K(X.Y)=(X.Y+c)^p$

در فرمول فوق p یک مقدار صحیح مثبت و $c\in\mathbb{R}$.

- Gaussian (RBF) kernel: $K(X.Y)=\exp(-\frac{\|X-Y\|^2}{2\sigma^2})$

در فرمول فوق $\sigma\in\mathbb{R}$ هسته گاووسی نامیده می‌شود.

- Sigmoidal kernel: $K(X.Y)=\tanh(a(X.Y)+b)$

در فرمول فوق $a.b\in\mathbb{R}$ هستند.

برای دو نقطه دلخواه X_i و Y_j در یک فرایند خوشه بندی مبتنی بر تکرار، نیاز به $K(X_i.Y_j)$ خواهیم داشت. بنابراین یک ماتریس به نام ماتریس Kernel، $H=[K_{ij}]_{n\times n}$ تعریف می‌شود که در آن $K_{ij}=K(X_i.Y_j)$ و n اندازه دیتاست است. ماتریس Kernel از قبل محاسبه شده و ذخیره می‌شود.

فرض می‌کنیم دیتاست با اندازه n برابر $D=\{X_1.X_2.\dots.X_n\}$ است. و k تعداد خوشه‌ها است و $M^{(0)}$ نقاط اولیه مرکز هر خوشه به صورت $M^{(0)}=\{m_1^{(0)}.m_2^{(0)}.\dots.m_k^{(0)}\}$ است که در آن هر نقطه در فضای ویژگی تعریف شده است. برای هر نمونه $X\in D$ ، الگوریتم X را به میانگین $m_i^{(0)}$ تخصیص می‌دهد به طوری که $m_i^{(0)}$ نزدیک‌ترین فاصله از X در فضای ویژگی القا شده Kernel[۸] است. نتایج اولیه پارتیشن بندی به صورت $\Pi^{(0)}$ تعریف شده است. پارامترهای k و D و $\Pi^{(0)}$ به عنوان پارامترهای ورودی به الگوریتم Kernel K means داده می‌شود و خروجی الگوریتم Π_D ، خوشه بندی کل دیتاست خواهد بود.

هدف این الگوریتم مینیمم کردن تابع هدف[۱۰] زیر است:

$$J=\sum_{j=1}^{K}\sum_{X_i\in C_j}\|\phi(X_i)-\mu_k\|^2 \quad (۲)$$

در فرمول فوق μ_k میانگین خوشه C_k است.

$$\mu_k=\sum_{X_i\in C_k}\frac{\phi(X_i)}{|C_k|} \quad (۳)$$

فاصله بین دو نقطه $\phi(X_i)$ و $\phi(X_j)$ در فضای القا شده برابر:

$$\begin{aligned}&\|\emptyset(X_i)-\emptyset(Y_j)\|^2\\&=\emptyset(X_i)^2-2\emptyset(X_i).\emptyset(X_j)+\emptyset(X_j)^2\\&=K(X_i.y_j)-2K(X_i.y_j)+K(X_i.y_j)\end{aligned} \quad (۴)$$

مقدار $\|\emptyset(X_i)-\mu_k\|^2$ را می‌توان بدون در نظر گرفتن تبدیل $\emptyset(.)$ به صورت صریح بدست آورد:

$$\begin{aligned}&\|\emptyset(X_i)-\mu_j\|^2\\&=\left\|\emptyset(X_i)-\sum_{X_i\in C_j}\frac{\phi(X_i)}{|C_k|}\right\|^2\\&=\emptyset(X_i).\emptyset(X_i)-F(X_i.C_k)+G(C_k)\end{aligned} \quad (۵)$$

که در فرمول فوق:

$$F(X_i.C_k)=-\frac{2}{|C_k|}\sum_{X_l\in C_k}\emptyset(X_l).\emptyset(X_i) \quad (۶)$$

$$G(C_k)=\frac{1}{|C_k|^2}\sum_{X_l\in C_k}\sum_{X_s\in C_k}\emptyset(X_l).\emptyset(X_s) \quad (۷)$$

متد تکراری این الگوریتم در الگوریتم ۱ نمایش داده شده‌است. پیچیدگی زمانی و مکانی این الگوریتم $O(n^2)$ است.

این الگوریتم وابستگی خیلی زیادی به انتخاب مناسب $\Pi^{(0)}$ دارد. یک انتخاب نامناسب برای $\Pi^{(0)}$ موجب خوشه بندی خیلی ضعیف می‌شود.

الگوریتم ۱ Kernel K means clustering method (D,K, $\Pi^{(0)}$)

1.For each cluster C_j , find $|C_j|$ and $G(C_j)$.
2.compute $F(X_i.C_j)$ for each X_i and each cluster C_j.
3.find $\|\emptyset(X_i)-\mu_j\|^2$ using equation (۴) and assign X_i until its nearest center.
4.update μ_j , for j=1 to k , using equation (۲)
5.repeat step 1 through step 4 till convergence.
Output:
The final partition $\Pi_D=\{C_1.C_2.\dots.C_K\}$

۳- روش پیشنهادی

برای حل حساسیت الگوریتم K means به انتخاب نقاط اولیه و مشکل مربوط به داده‌های غیرخطی و واگرا، روشی بر پایه تکرار که باعث افزایش کیفیت پارتیشن بندی به طور سیستماتک می‌شود ارائه کرده‌ایم. در این بخش به تشریح اجزای الگوریتم می‌پردازیم.

۳-۱- تابع هدف ماکزیمم واریانس

با در نظر گرفتن $\chi=\{\emptyset(X_i)\}_{i=1}^{N}$ به عنوان دیتاست به طوری که $x_i\epsilon\mathcal{R}^d$، و تقسیم آن به M خوشه جدا از هم به طوری که مرکز هر خوشه μ_j است، ما به جای مینیمم کردن مجموع واریانس داخل خوشه ای (۱) سعی می‌کنیم ماکزیمم واریانس داخل خوشه‌ای را مینیمم کنیم.[۱۱]

$$\varepsilon_{max}=\max_{1\le k\le M}V_K=\max_{1\le k\le M}\{\delta_{ik}\|\emptyset(X_i)-\mu_K\|^2\} \quad (۸)$$

که در آن مقادیر V_K و μ_K و δ_{ik} قبلا توضیح داده شده است.

منطق این رویکرد به صورت زیر است: مجموع واریانس تمامی خوش‌ها در تابع هدف K means یعنی معادله (۱) که در مبحث k means بحث شد، اجازه می‌دهد چند خوشه با مقدار واریانس زیاد با تعداد زیادی خوشه با واریانس کم، ترکیب شود. یعنی تعادل نسبی درمیان واریانس خوشه‌ها در روش K means وجود ندارد. در حالی که واریانس خوشه ها معیاری از کیفیت خوشه بندی است. مینیمم کردن ماکزیمم واریانس، از به وجود آمدن خوشه‌هایی با واریانس بالا جلوگیری می‌کند و در نهایت خوشه‌هایی به دست می‌آید که واریانس شبیه به هم دارند. رویکرد بالا دو دست آورد مهم دارد: چون K means ، ε_{sum} را مینیم

میکند تفاوتی بین دو خوشه قائل نمی‌شود، فقط سعی می‌کند مجموع واریانس‌ها را کاهش دهد. بنابراین انتخاب نقاط اولیه نامناسب موجب یک راه حل ضعیف می‌شود که در میان خوشه‌ها تفاوت فاحشی در واریانس خوشه‌ها وجود دارد و ممکن است خوشه ای با واریانس کم به خوشه‌های کوچکتر تقسیم شود و یا خوشه‌هایی با واریانس بیشتر با یکدیگر ادغام شوند و یا خوشه‌هایی با داده‌های پرت به وجود آیند. ولی در تابع هدف ε_{max} کمتر احتمال دارد که به یک مقدار مثل روش بالا همگرا شود به عبارتی غلبه کردن برای مشکل نقاط اولیه در این روش آسان است. بنابراین روش K means با تابع ε_{sum} قادر به پوشش دادن بهتر داده‌ها در خوشه‌ها در چندین تکرار نیست. یک مثال در شکل (۲) نمایش داده شده است.

۲-۳- تابع هدف نرمال شده

علیرغم مزیت‌های فوق، مینیمم کردن ماکزیمم واریانس خوشه، یک مشکل بهینه سازی غیربدهی را نشان می‌دهد. برای حل این مشکل تابع هدف را نرمال سازی می‌کنیم. به طوری که بتوان بهینه سازی را به راحتی در فاز تکرار kernel k means انجام داد. برای مجموع واریانس داخل خوشه‌ای یک راهکار بر اساس وزن دهی ε_w ایجاد کردیم(۹) که تاکید بر روی دادن وزن بیشتر به خوشه‌ای است که واریانس آن بیشتر است. هدف از این کار تقلید رفتاری است که ε_{max} از خود نشان می‌دهد.

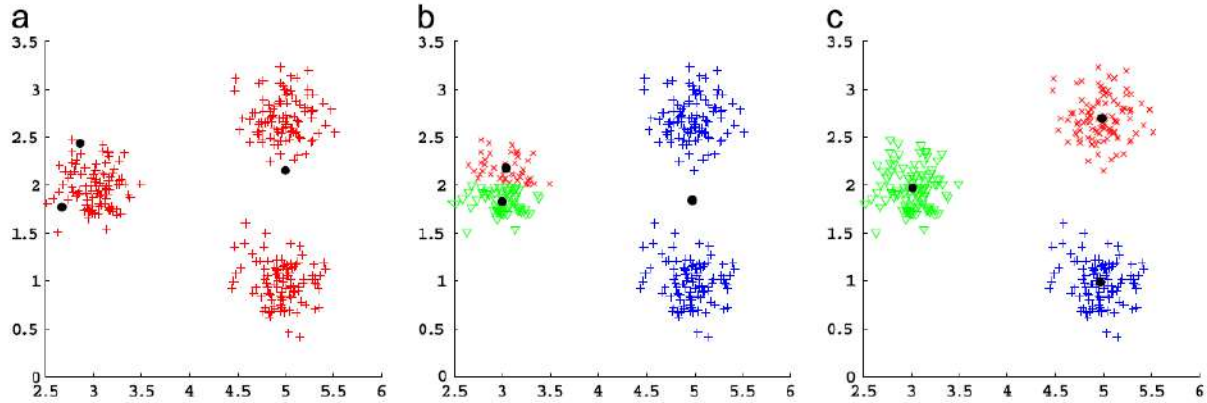
$$\varepsilon_w \sum_{k=1}^{M} W_K^P V_K \quad \sum_{k=1}^{M} W_K^P \sum_{i=1}^{N} \delta_{ik} \|\emptyset(X_i) - \mu_K\|^2 \qquad (۹)$$
$$W_K \geq 0 \;.\; \sum_{K=1}^{M} w_K = 1 \qquad 0 \leq p < 1$$

بر خلاف ε_{max} حالا تمامی خوشه‌ها با یک هدف گسترش یافته‌اند، اگرچه با درجه‌های مختلفی از w_k تنظیم شده‌اند. به طور قابل مشاهده ای بیش از یک خوشه (با وزن بیشتر) تلاش می‌کند که واریانس آن به حداقل برسد. در این مسیر کاهش بزرگتر از هدف نیز امکان پذیر است. باید خاطر نشان کرد که وزن‌ها ثابت نیستند اما پارامترها باید با برچسب خوشه‌ها بهینه شوند. وزن‌ها را به عنوان پارامترهایی که منعکس کننده‌ی واریانس خوشه‌ها در هر تکرار آموزش هستند، تعریف می‌کنیم. در واقع این مجموعه‌ی واحد سعی می‌کند over-fitting و مشکل بهینه سازی را حل کند. توان P به وسیله کاربر انتخاب می‌شود که در بازه (۱و۰] است و در واقع کنترل حساسیت آپدیت وزن‌ها نسبت به تفاوت واریانس خوشه‌ها را نشان می‌دهد. هر چه قدر این مقدار بیشتر باشد یعنی تاثیر وزن بیشتر از تاثیر واریانس آن خوشه است. در بخش بعدی جزئیات بیشتر در مورد p را مورد بحث قرار خواهیم داد.

هدف از این خوشه بندی تولید خوشه‌هایی با واریانس داخل خوشه‌ای کمتر و همزمان یک گارد محکمی در برابر تولید خوشه‌هایی با واریانس بالاست. به عبارتی هدف از نرمال سازی، تنبیه کردن خوشه‌های بزرگ است. در واقع منظور از تنبیه کردن، اختصاص دادن وزن بیشتر به خوشه‌ای است که واریانس آن بالاست. که این کار با یافتن ماکزیمم مقدار برای ε_w امکان پذیر است. در نتیجه می‌توان فرم کلی روش Min-Max Kernel K means را به صورت زیر نوشت:

$$\min_{\{C_K\}_{K=1}^{M}} \min_{\{w_K\}_{K=1}^{M}} \varepsilon_w \qquad (۱۰)$$
$$W_K \geq 0 \;.\; \sum_{K=1}^{M} w_k = 1 \;.\; 0 \leq p < 1$$

ما الگوریتمی بر پایه تکرار که آن را Min-Max Kernel K means، می نامیم ارائه کرده‌ایم که الگوریتم بین C_K و w_K به طور متناوب تکرار می‌شود تا به یک مقدار بهینه ε_w برسیم. جزئیات بیشتر در مورد مشتقات این روش در بخش بعد توضیح داده می‌شود.



شکل (۲) (a) انتخاب نقاط مرکزی نامناسب (b) روش K means منجر به راه حل ضعیف می‌شود، شامل خوشه‌هایی با واریانس‌های مختلف. روش ارائه شده بر اساس ε_{max} در (c) نمایش داده شده است که از همان نقاط مرکزی شروع شده است. سمبل‌ها و رنگ‌های مختلف نشان دهنده تخصیصات به خوشه‌ها هستند.

۳-۲-۱- گام ماکزیمم سازی

با ثابت نگه داشتن وزن، تخصیصات به خوشه جدید و μ_j محاسبه می شود. برای اختصاص دادن نقاط به هر خوشه، تعریف δ_{ik} به‌صورت زیر تغییر می‌کند (۱۱):

$$\delta_{ik} = \begin{cases} 1. & k = \text{argmin}_{1\le k\le M} W_K^P \|\emptyset(X_i) - \mu_K\|^2 \\ 0. & \text{otherwise} \end{cases} \quad (11)$$

این کا برای بهینه سازی δ_{ik} انجام می‌شود. در واقع هر نمونه به خوشه-ای اختصاص می‌یابد که فاصله-وزن آن از مرکز خوشه‌ها کمترین باشد. برا تخمین مقدار μ_K که مستقل از وزن خوشه هاست داریم (۱۲):

$$\mu_K = \frac{\sum_1^N \delta_{ij}\emptyset(X_i)}{\sum_1^N \delta_{ik}} \quad (12)$$

۳-۲-۲- گام مینیم سازی

برای بروز کردن وزن‌ها برای تخصیصات و مرکز خوشه‌ها در معادلعه (۱۱) ضریب لاگرانژ و مشتقات آن را با تنظیم کردن وزن‌ها به صفر در ابتدا با هم متحد کردیم. در واقع بعد چندین دستکاری با توجه به مقدار p فرمول زیر حاصل شد:

$$w_K = V_K^{\frac{1}{1-P}} / \sum_{K=1}^{M} V_{\acute{K}}^{\frac{1}{1-P}} \quad (13)$$

$$v_k = \sum_{i=1}^{N} \delta_{ik} \|x_i - \mu_k\|^2$$

چون $1/(1-P) > 0$ و $0 \le P < 1$ مشاده می‌شود که وزن بیشتر به خوشه ای که واریانس بیشتر دارد اختصاص می‌یابد.

۳-۳- بحث در مورد پارمترهای الگوریتم

انتخاب مقدار مناسب برای p ، باعث افزایش کیفیت خوشه بندی می شود. اگر مقدار p نزدیک به یک انتخاب شود تاثیر وزن بر خوشه بندی زیاد است و خوشه بندی درستی به دست نمی‌آید. اگر مقدارp صفر انتخاب شود عملا الگوریتم تبدیل به الگوریتم Kernel K means می شود اگر مقدار P را دقیقا یک تعریف کنیم در این صورت وزن‌ها به صورت زیر تغییر می‌کند:

$$W_K = \begin{cases} 1. & K = \text{argmax}_{1\le \acute{k} \ll M} V_{\acute{K}} \\ 0. & \text{otherwise} \end{cases} \quad (14)$$

مشاهده می‌شود که وقتی p=1 باشد، در هر تکرار بزرگترین واریانس خوشه یک وزن غیر صفر دریافت می‌کند و بنابراین در مرحله مینیمم سازی همه‌ی نمونه‌ها به صورت تصادفی انتخاب می‌شوند.

پس مقدار مناسب برای p در بازه $0\le p<1$ است. بهترین مقدار برای p یک مقدار حد متوسط است. مقدار دقیق این حد متوسط مشخص نیست. برای پیدا کردن این مقدار متوسط از یک الگوریتمی بر پایه تکرار استفاده می‌کنیم که در بخش بعد توضیح داده شده است.

در Min-Max Kernel K means ابتدا باید الگوریتم به وزن‌ها و تخصیصات دسترسی داشته باشد و چون وزن در طی اجرای الگوریتم در تکرارها آپدیت می‌شود، مقدار اولیه برای وزن‌ها را به صورت $w_K = \frac{1}{M}$ تعریف می‌کنیم.

۳-۴- روش بهبود یافته برای بدست آوردن پارامترها

با توجه به مباحث فوق، مقدار مناسب برای p مقدر حد متوسط است. بر این اساس برای پیدا کردن مقدار دقیق p از یک الگوریتمی بر پایه تکرار استفاده می‌کنیم که در طی اجرای این الگوریتم مقدار دقیق p به دست می‌آید. در این رویکرد از ۳ مقدار p_{init} و p_{step} و p_{max} استفاده می‌کنیم. مقدار p_{init} را با صفر (مقداری اختیاری است ولی کوچکترین مقدار p صفر است و معمولا صفر در نظر می‌گیرند) نشان می‌دهیم. p_{step} نشان دهنده مقدار افزایش p است. در واقع در هر تکرار مقدار p به صورت $p + p_{step}$ افزایش می‌یابد. p_{max} نشانگر نهایت مقدار برای p است که نباید یک یا نزدیک به یک انتخاب شود چون اگر یک انتخاب شود الگوریتم وابستگی شدیدی به وزن‌ها پیدا می‌کند. الگوریتم با مقدار کوچک p_{init} شروع شده و در هر تکرار به اندازه p_{step} افزایش می‌یابد این افزایش تا زمانی ادامه می‌یابد که p_{max} به دست آید. گام اول خوشه بندی به شدت از مقادیر اولیه تاثیر می‌پذیرد بنابرایم مقدار مناسب برای p، p کوچک است ($p = p_{init}$). با پیشرفت خوشه بندی، p به طور آهسته افزایش می‌یابد و از به وجود آمدن خوشه‌هایی با واریانس بالا جلوگیری می‌کند در مراحل اولیه افزایش، همچنان خروجی خوشه بندی مناسب نیست. باید توجه داشت که تنبیه خوشه‌ها با واریانس بالا تا زمانی که مقدار p ثابت نباشد عملی نیست. اگر در طی این افزایش‌ها خوشه‌ای خالی و یا خوشه‌ای با یک نمونه ظاهر شد، آن خوشه در مرحله ماکزیمم سازی، وزن صفر می‌گیرد و بدون توجه به این که آیا در مرحله افزایش، p به p_{max} رسیده یا نه، p مقدار p قبلی را بر می‌گرداند و سپس الگوریتم تخصیصات مربوط به تکرار قبلی را بر می‌گرداند.

برای اینکه این الگوریتم پایدار باشد، یک مقدار با عنوان پارامتر تاثیر گزاری حافظه‌به[۲] الگوریتم اضافه می‌شود:

$$w_k^{(t)} = \beta w_k^{(t-1)} + (1-\beta)(V_K^{\frac{1}{1-P}} / \sum_{K=1}^{M} V_{\acute{K}}^{\frac{1}{1-P}}) \quad (15)$$
$$0 \le \beta \le 1$$

در واقع β تاثیر تکرار وزن‌های قبلی به آپدیت‌های جاری را کنترل می-کند. تغییر مقدار نرمال سازی (۹) بین دو تکرار متوالی، اگر از ϵ (یک مقدار کوچک)کوچکتر باشد، الگوریتم پایان می‌پذیرد. اگر همگرایی به یک مقدار اتفاق نیافتاد، نهایتا با یک مقدار از پیش تعیین شده برای تکرار (t_{max}) الگوریتم پایان می‌پذیرد. در عمل مشاهده کردیم همگرایی قبل از رسیدن به t_{max} اتفاق می‌افتد. شبه کد این روش در الگوریتم ۲ نشان داده شده است.

الگوریتم ۲ Min-Max Kernel K means

Inpute: dataset $\chi = \{x_i\}_{i=1}^{N}$, Initial centers $\{\mu_k^{(0)}\}_{K=1}^{M}$, Number of clusters M, Secondary parameters $t_{max}\,.p_{max}.p_{step}.\epsilon.\beta$

set t=0
set $p_{init} = 0$
set $w_k^{(0)} = \frac{1}{M}.\ \forall K = 1\,.\ \ldots.\ M$
set empty= false //No empty or singleton clusters yet detected
$p = p_{init}$
repeat
t=t+1
for i=1 to N do //update the cluster assignments
for k=1 to M do

$$\delta_{ik} = \begin{cases} 1. & k = argmin_{1\le k\le M}(W_k^{(t-1)})^p \left\|\emptyset(X_i) - \mu_K^{(t-1)}\right\|^2 \\ 0. & otherwise \end{cases}$$

end for
end for
if empty od singleton clusters have occurred at time t then //reduce p.
empty=true
$p = p - p_{step}$
if p < p_{init} then
return NULL
end if
//Revert to the assignments and weights corresponding to the reduce p.
$\delta_{ik}^{(t)} = [\Delta^{(p)}]_{ik}.\forall K = 1\,.\ \ldots.\ M\,.\ \forall i = 1\,.\ \ldots.\ N$
$W_k^{(t-1)} = [W^{(p)}]_k.\forall K = 1\,.\ \ldots.\ M$
end if
for all μ_k , k=1 to M
$\mu_K = \frac{\sum_1^N \delta_{ij}\emptyset(X_i)}{\sum_1^N \delta_{ik}}$
end for
if p < p_{max} and empty=false then //increase p.
$\Delta^{(p)} = [\delta_{ik}^{(t)}]$ //store the current assignment in matrix $\Delta^{(p)}$.
$w^{(p)} = [W_k^{(t-1)}]$ // store the previous weights in vector $w^{(p)}$
$p = p + p_{step}$
end if
for all w_k , k=1 to M do // update the weights.
$w_k^{(t)} = \beta w_k^{(t-1)} + (1-\beta)(V_K^{1/(1-P)}/\sum_{\hat{K}=1}^{M} V_{\hat{K}}^{1/(1-P)})$, where $v_k = \sum_{i=1}^{N} \delta_{ik}\|x_i - \mu_k\|^2$
end for
until $\left|\varepsilon_w^{(t)} - \varepsilon_w^{(t-1)}\right| < \epsilon\ or\ t \ge t_{max}$
return $\{\delta_{ik}^{(t)}\}_{i=1,\ldots,N\,,K=1,\ldots,M\,,\{\mu_k^{(t)}\}_{K=1}^{M}}$

۴- نتایج آزمایشات

در این بخش یک ارزیابی از الگوریتم پیشنهادی در مقایسه با سایر الگوریتم‌ها بر روی ده مجموعه داده واقعی از انبار داده UCI انجام شد [20]. اطلاعات مربوط به این دیتاست‌ها در جدول (۱) خلاصه شده است.

جدول (۱) دیتاست انتخاب شده از UCI

دیتاست	نمونه	صفت	خوشه
Ecoli	۳۳۶	۸	۸
Coil2	۲۱۶	۱۰۰۰	۳
Wine	۱۷۸	۱۳	۳
Iris	۱۵۰	۴	۳
Bupa	۳۴۵	۷	۲
Glass	۲۱۴	۱۰	۶
CMC	۱۴۷۳	۹	۳
Thyroid	۲۱۵	۶	۳
Wisconsin breast cancer	۵۶۹	۳۲	۲
Vowel	۸۷۱	۳	۶

درآزمایشات زیر مقادر p_max=0.5 و p_step=0.01 و p_init=0 و t_max=500 در نظر گرفته شده است. در روش پیشنهادی و الگوریتم Kernel K means برای σ سه مقدار ۱ و ۵۰ و ۱۰۰ در نظر گرفته شده است. برای بتا (تاثیر گذاری حافظه) سه مقدار ۰ و ۰،۵ و ۱ در نظر گرفته

شده است. این مقادیر بر روی نتایج نهایی تاثیر کمی داشته اند. الگوریتم برای هر یک از روش‌ها ده بار تکرار شده است و میانگین آنها در ده بار تکرار ثبت شده است. از Silhouette و مجموع مربع خطا به عنوان معیار ارزیابی استفاده شده است.

با توجه به آزمایشات انجام شده الگوریتم ارائه شده بهتر از سایر الگوریتم-ها عمل می‌کند ولی در بعضی از دیتاست‌ها الگوریتم ارائه شده از نظر معیار Silhouette کارایی کمتری دارد. در میان تمامی الگوریتم‌ها مجموع واریانس و بیشینه واریانس برای Min-Max Kernel K means کمترین مقدارها را دارند که این نشان دهنده کیفیت بالای خوشه بندی است. معیار SSE برای تمامی حالات بهترین عملکرد را دارد. در ادامه به بررسی هر یک از دیتاست ها می‌پردازیم.

در دیتاست Ecoli همان طور که در جدول (۲) مشاهده می‌شود زمانی که σ برابر ۵۰ است، الگوریتم ارائه شده بیشترین کارایی را دارد. بطور دقیق می‌توان گفت الگوریتم ارائه شده برای این دیتاست زمانی که بتا برابر ۰٫۵ و σ برابر ۵۰ است، بیشترین کارایی را دارد. در این میان الگوریتم Min-Max K means بدترین کارایی را دارد و قادر به خوشه بندی دیتاست با نقاط انتخاب شده نیست. معیار SSE نشان می‌دهد روش ارائه شده در مقیاسه با سایر الگوریتم‌ها بهتر عمل می‌کند. به طور کلی برای این دیتاست می‌توان گفت الگوریتم ارائه شده بهترین عملکرد را در میان سایر الگوریتم‌ها دارد.

جدول (۲) دیتاست Ecoli

	Sum of variance	Max of variance	Silhouette coefficient	SSE
Min-Max K means (β=0)	NAN	NAN	NAN	NAN
Min-Max K means (β=0.5)	NAN	NAN	NAN	NAN
Min-Max K means (β=1)	NAN	NAN	NAN	NAN
Traditional K means	۱۳,۷۳۹۵	۵,۷۹۵۱	۰,۲۳۰۸	۱۶۶,۵۲۸۹
σ=1				
Min-Max Kernel K means (β=0)	NAN	NAN	NAN	NAN
Min-Max Kernel K means (β=0.5)	۲,۲۴۴۰	۰,۹۶۲۶	۰,۲۳۳۸	۶۳,۷۰۳۰
Min-Max Kernel K means (β=1)	۲,۲۰۷۰	۰,۹۵۷۳	۰,۲۲۹۷	۶۴,۲۲۹۷
Kernel K means	۶,۷۴۳۶	۲,۷۳۲۹	۰,۲۲۹۷	۱۰۸,۲۱۴۷
σ=50				
Min-Max Kernel K means (β=0)	NAN	NAN	NAN	NAN
Min-Max Kernel K means (β=0.5)	۲,۲۷۴۳	۰,۹۶۶۸	۰,۲۳۶۰	۶۳,۶۲۳۰
Min-Max Kernel K means (β=1)	۲,۲۲۰۹	۰,۹۶۶۵	۰,۲۲۸۱	۶۴,۳۱۹۱
Kernel K means	۶,۷۵۶۰	۲,۷۶۲۰	۰,۲۲۸۱	۱۰۷,۹۲۴۱
σ=100				
Min-Max Kernel K means (β=0)	NAN	NAN	NAN	NAN
Min-Max Kernel K means (β=0.5)	NAN	NAN	NAN	NAN
Min-Max Kernel K means (β=1)	۲,۲۲۰۹	۰,۹۶۶۵	۰,۲۲۸۱	۶۴,۳۱۹۱
Kernel K means	۶,۷۵۶۰	۲,۷۶۲۰	۰,۲۲۸۱	۱۰۷,۹۲۴۱

برای دیتاست coil2، با توجه به جدول (۳) الگوریتم ارائه شده عملکرد بهتری نسبت به سایر الگوریتم‌ها از خود نشان می‌دهد. در این دیتاست الگوریتم زمانی که بتا برابر ۰ و σ برابر ۵۰ است بهترین کارایی را دارد.

جدول (۳) دیتاست Coil2

	Sum of variance	Max of variance	Silhouette coefficient	SSE
Min-Max K means (β=0)	۵,۶۹۷۹۱۱	۲,۸۸۸۵۴۵	۰,۱۱۸۷۲۶	۵۵۳,۶۴۶۷۷۱
Min-Max K means (β=0.5)	۵,۷۱۱۳۲۹	۲,۸۹۹۳۷۵	۰,۱۱۷۶۵۴	۵۵۳,۹۰۶۳۱۳
Min-Max K means (β=1)	۵,۹۶۱۷۸۴	۳,۰۰۱۴۷۱	۰,۱۱۶۲۶۲	۵۵۳,۶۴۶۷۷۱
Traditional K means	۱۱,۵۹۲۰	۵,۸۸۲۵	۰,۱۰۴۷	۲۶۷,۵۴۷۶

σ=1				
Min-Max Kernel K means (β=0)	۴,۶۵۹۶۶۶	۲,۴۱۵۶	۰,۱۱۷۳	۱۸۱,۴۲۱۵
Min-Max Kernel K means (β=0.5)	۴,۶۶۱۶	۲,۴۱۵۵	۰,۱۱۷۳	۱۸۱,۴۲۰۴
Min-Max Kernel K means (β=1)	۴,۸۱۵۲	۲,۵۱۴۲	۰,۱۱۵۳	۱۸۱,۵۳۳۳
Kernel K means	۱۰,۳۸۴۷	۵,۳۹۹۶	۰,۱۱۵۳	۲۶۲,۶۷۷۱
σ=50				
Min-Max Kernel K means (β=0)	۴,۶۷۴۷	۲,۴۲۱۳	۰,۱۱۸۷	۱۸۱,۲۵۶۰
Min-Max Kernel K means (β=0.5)	۴,۶۷۴۴	۲,۴۳۱۹	۰,۱۱۸۷	۱۸۱,۲۵۰۶
Min-Max Kernel K means (β=1)	۴,۶۷۹۰	۲,۵۰۰۴	۰,۱۱۶۳	۱۸۱,۵۲۶۸
Kernel K means	۱۰,۲۹۱۹	۵,۳۸۳۴	۰,۱۱۶۳	۲۶۲,۹۵۰۵
σ=100				
Min-Max Kernel K means (β=0)	۴,۶۷۴۷	۲,۴۲۱۳	۰,۱۱۸۷	۱۸۱,۲۵۶۰
Min-Max Kernel K means (β=0.5)	۴,۶۸۴۴	۲,۴۳۱۹	۰,۱۱۸۷	۱۸۱,۲۵۰۶
Min-Max Kernel K means (β=1)	۴,۷۶۹۰	۲,۵۰۰۴	۰,۱۱۶۳	۱۸۱,۵۲۶۸
Kernel K means	۱۰,۲۹۱۹	۵,۳۸۳۴	۰,۱۱۶۳	۲۶۲,۹۵۰۵

برای دیتاست glass جدول (۴) با توجه به معیار Silhouette الگوریتم عملکرد خوبی نداشته است. در این دیتاست بهترین کارایی را الگوریتم K means داشته است. لازم به ذکر است که معیار SSE برای الگوریتم ارائه شده بهتر از سایر الگوریتم‌ها است. با افزایش مقدار σ، کارایی الگوریتم از نظر دو معیار SSE و Silhouette بیشتر می‌شود. ولی مقدار ماکزیمم واریانس و مجموع واریانس بشدت افزایش می یابد که نشان دهنده‌ی کیفیت پایین خوشه بندی است.

جدول (۴) دیتاست Glass

	Sum of variance	Max of variance	Silhouette coefficient	SSE
Min-Max K means (β=0)	۳۱۲,۷۹۱۷۳۶	۱۰۵,۸۸۲۲۳۴	۰,۳۶۸۳۲۰	۲۴۳۰,۴۸۲۵۶۸
Min-Max K means (β=0.5)	۳۰۱,۱۷۳۹۸۲	۹۷,۹۱۵۹۳۰	۰,۳۶۷۸۵۰	۲۴۰۸,۵۰۵۳۸۸
Min-Max K means (β=1)	۲۶۵,۴۳۹۱۴۴	۹۲,۱۸۸۸۳۶	۰,۳۰۹۸۱۱	۲۲۱۵,۷۰۴۵۵۰
Traditional K means	۴۹۰,۱۰۰۷	۲۱۷,۰۴۳۴	۰,۳۷۸۹	۴۸۱,۲۹۷۸
σ=1				
Min-Max Kernel K means (β=0)	۴۸,۱۵۵۹	۲۲,۸۶۱۵	۰,۱۷۱۸	۲۵۵,۹۷۵۲
Min-Max Kernel K means (β=0.5)	۴۸,۱۲۳۱	۲۳,۰۴۱۶	۰,۱۷۰۴	۲۵۶,۷۰۱۱
Min-Max Kernel K means (β=1)	۶۲,۸۶۵۸	۳۱,۶۳۲۴	۰,۱۶۶۸	۲۵۷,۸۶۳۵
Kernel K means	۱۲۸,۱۲۷۰	۶۰,۰۷۴۱	۰,۱۶۸۷	۳۷۹,۷۹۸۳
σ=50				
Min-Max Kernel K means (β=0)	۱۰۰,۷۷۰۸	۵۷,۵۹۳۸	۰,۳۲۰۵	۲۲۲,۸۶۱۴
Min-Max Kernel K means (β=0.5)	۱۰۰,۰۳۲۵	۵۶,۶۰۵۲	۰,۳۱۸۷	۲۲۲,۶۵۵۲
Min-Max Kernel K means (β=1)	۹۲,۵۸۳۷	۵۲,۵۰۲۲	۰,۳۰۹۸	۲۲۲,۸۹۵۳
Kernel K means	۳۳۷,۶۹۸۸	۱۹۲,۷۰۷۶	۰,۳۰۹۸	۳۹۸,۰۶۳۲
σ=100				
Min-Max Kernel K means (β=0)	۱۰۰,۷۷۰۸	۵۷,۵۹۳۸	۰,۳۲۰۵	۲۲۲,۸۶۱۴
Min-Max Kernel K means (β=0.5)	۱۰۰,۰۳۲۵	۵۶,۶۰۵۲	۰,۳۱۸۷	۲۲۲,۶۵۵۲
Min-Max Kernel K means (β=1)	۹۲,۵۸۳۷	۵۲,۵۰۲۲	۰,۳۰۹۸	۲۲۲,۸۹۵۳
Kernel K means	۳۳۷,۶۹۸۸	۱۹۲,۷۰۷۶	۰,۳۰۹۸	۳۹۸,۰۶۳۲

دیتاست Bupa با توجه به جدول (۵) از معیار SSE پایین‌تری نسبت به بقیه برخوردار است ولی در معیار Silhouette بدترین کارایی را دارد مقدار ماکزیمم واریانس و مجموع واریانس خوشه‌ای برای این دیتاست پایین‌تر از سایر الگوریتم‌ها است.

جدول (۵) دیتاست Bupa

	Sum of variance	Max of variance	Silhouette coefficient	SSE
Min-Max K means (β=0)	۱۷۸۲۲,۴۶۴۱۱۹	۱۲۴۰۶,۱۸۴۳۵۷	۰,۶۱۴۶۹۷	۴۲۵۲۸,۴۴۸۷۹۲
Min-Max K means (β=0.5)	۱۷۵۳۲,۵۳۷۸۲۹	۱۲۱۹۹,۵۶۴۳۷۳	۰,۶۱۱۴۰۶	۴۲۲۰۳,۲۰۴۹۵۸
Min-Max K means (β=1)	۱۹۹۶۹,۵۵۹۴۴۰	۱۳۹۲۱,۸۴۱۴۹۰	۰,۶۳۳۱۸۲	۴۴۸۴۶,۴۲۴۶۰۴
Traditional K means	۹۴۵۷۸	۸۵۷۰۰	۰,۶۳۱۰	۲۲۴۳۲
σ=1				
Min-Max Kernel K means (β=0)	۹۷۹۶,۶	۸۲۱۰,۲	۰,۳۰۵۷۰۵	.۱۲۳۴۷
Min-Max Kernel K means (β=0.5)	۹۷۹۶,۶	۸۲۱۰,۲	۰,۳۰۵۷۰۵	.۱۲۳۴۷
Min-Max Kernel K means (β=1)	۹۷۹۷۹۶,۶	۸۲۸۲۱۰,۲	۰,۳۰۵۷۰۵	۱۲۳۴۷
Kernel K means	۱۰۴۳۷	۸۷۷۵,۱۱	۰,۳۰۵۷	۱۲۷۵۷
σ=50				
Min-Max Kernel K means (β=0)	۶۰۸۹,۶	۴۸۷۸,۲	۰,۴۹۳۱	۱۰۰۷۲
Min-Max Kernel K means (β=0.5)	۶۰۷۸,۷	۴۸۶۸,۹	۰,۴۹۳۰	۱۰۰۷۱
Min-Max Kernel K means (β=1)	۵۷۴۱,۴	۴۶۲۷,۶	۰,۴۴۰۲	۱۰۳۲۴
Kernel K means	۱۴۶۲۸	۱۱۶۱۳	۰,۴۴۰۲	۱۴۵۳۰
σ=100				
Min-Max Kernel K means (β=0)	۷۰۸۰,۸	۶۰۶۱,۳	۰,۵۶۷۲	۹۹۹۸,۲
Min-Max Kernel K means (β=0.5)	۷۰۸۰,۸	۶۰۶۱,۳	۰,۵۶۷۲	۹۹۹۸,۲
Min-Max Kernel K means (β=1)	۷۰۸۰,۸	۶۰۶۱,۳	۰,۵۶۷۲	۹۹۹۸,۲
Kernel K means	۱۸۴۶۱	۱۵۸۹۱	۰,۵۶۷۲	۱۴۷۳۱

برای دیتاست wine در جدول (۶) الگوریتم ارائه شده بهترین کارایی را دارد. برای این دیتاست، الگوریتم ارائه شده از لحاظ معیار SSE، ماکزیمم واریانس و مجموع واریانس بسیار خوب عمل می‌کند.

جدول (۶) دیتاست Wine

	Sum of variance	Max of variance	Silhouette coefficient	SSE
Min-Max K means (β=0)	۱۱۷۵۷۲۸,۵۳۷۴۰۷	۴۸۵۷۱۵,۷۷۶۰۳۲	۰,۵۴۷۳۹۵	۱۴۳۳۲۰,۹۳۹۸۳۸
Min-Max K means (β=0.5)	۱۱۴۶۷۸۶,۲۰۸۶۸۳	۴۶۹۸۴۶,۷۳۶۰۵۶	۰,۵۵۳۲۵۵	۱۴۰۰۵۵,۵۴۷۹۶۶
Min-Max K means (β=1)	۱۳۰۹۳۶۱,۹۶۱۸۳۳	۵۷۲۴۰۱,۹۵۴۲۸۹	۰,۵۶۶۵۱۶	۱۵۷۳۸۹,۲۸۶۶۷۲
Traditional K means	۲۷۱۶۷۰۰	۱۶۸۹۴۰۰	۰,۵۵۹۶	۷۶۳۴۹
σ=1				
Min-Max Kernel K means (β=0)	۶۰۷۴۰۰	۴۰۸۱۰۰	۰,۴۷۱۷	۳۰۸۲۲
Min-Max Kernel K means (β=0.5)	۶۰۷۴۰۰	۴۰۸۱۰۰	۰,۴۷۱۷	۳۰۸۲۲
Min-Max Kernel K means (β=1)	۶۰۷۴۰۰	۴۰۸۱۰۰	۰,۴۷۱۷	۳۰۸۲۲
Kernel K means	۶۰۷۴۰۰	۴۰۸۱۰۰	۰,۴۷۱۷	۳۰۸۲۲
σ=50				
Min-Max Kernel K means (β=0)	۲۷۸۶۴۰	۱۶۸۶۲۰	۰,۴۸۲۹	۲۲۰۵۸
Min-Max Kernel K means (β=0.5)	۲۷۷۵۰۰	۱۶۸۲۶۰	۰,۴۸۰۳	۲۲۰۹۳

Min-Max Kernel K means (β=1)	۲۷۷۱۷۰	۱۶۹۹۵۰	۰,۵۰۵۸	۲۰۶۵۲
Kernel K means	۵۱۲۲۳۰	۳۰۲۶۹۰	۰,۵۰۵۸	۳۱۳۷,۲
σ=100				
Min-Max Kernel K means (β=0)	۱۲۳۲۳۰	۵۱۰۸۳	۰,۵۴۳۶	۱۷۱۸۵
Min-Max Kernel K means (β=0.5)	۱۲۳۲۳۰	۵۱۰۸۳	۰,۵۴۳۶	۱۷۱۸۵
Min-Max Kernel K means (β=1)	۱۳۸۸۷	۶۳۶۷۰	۰,۵۴۹۸	۱۸۱۴۱
Kernel K means	۴۲۹۳۷۰	۲۱۷۵۰۰	۰,۵۴۹۸	۳۲۷۰۳

برای دیتاست iris با توجه به جدول (۷)، الگوریتم ارائه شده عملکرد نسبتا یکسانی با الگوریتم Min-Max k_means دارد. درمیان تمامی حالات مورد آزمایش، بهترین کارایی به ازای σ برابر ۱۰۰ و بتا برابر ۰,۵ است.

جدول (۷) دیتاست Iris

	Sum of variance	Max of variance	Silhouette coefficient	SSE
Min-Max K means (β=0)	۲۳,۸۸۷۸۸۸	۱۲,۷۷۶۳۵۸	۰,۵۴۵۸۵۱	۴۹۴,۵۵۸۵۵۰
Min-Max K means (β=0.5)	۲۳,۸۸۷۸۸۸	۱۲,۷۷۶۳۵۸	۰,۵۴۵۸۵۱	۴۹۴,۵۵۸۵۵۰
Min-Max K means (β=1)	۲۹,۲۶۲۲۹۹	۱۵,۶۷۱۰۰۸	۰,۵۴۴۱۶۴	۵۵۳,۳۳۴۸۴۸
Traditional K means	۲۲,۶۹۲۶	۱۶,۳۴۹۴	۰,۵۵۱۰	۲۳۵,۶۹۰۱
σ=1				
Min-Max Kernel K means (β=0)	۲,۶۲۴۰	۱,۵۳۲۹	۰,۵۲۹۹	۱۰۴,۱۶۸۳
Min-Max Kernel K means (β=0.5)	۲,۶۲۴۰	۱,۵۳۲۹	۰,۵۲۹۹	۱۰۴,۱۶۸۳
Min-Max Kernel K means (β=1)	۲,۵۷۰۵	۱,۵۰۲۲	۰,۵۲۹۴	۱۰۴,۳۰۸۸
Kernel K means	۱۷,۶۹۱۷	۱۱,۴۷۷۶	۰,۵۲۹۴	۲۳۴,۳۷۰۵
σ=50				
Min-Max Kernel K means (β=0)	۳,۱۰۶۵	۱,۸۴۴۰	۰,۵۴۴۳	۱۰۲,۶۷۰۹
Min-Max Kernel K means (β=0.5)	۳,۱۰۶۵	۱,۸۴۴۰	۰,۵۴۴۳	۱۰۲,۶۷۰۹
Min-Max Kernel K means (β=1)	۳,۰۲۳۸	۱,۸۱۸۷	۰,۵۴۴۲	۱۰۲,۱۹۵۱
Kernel K means	۲۱,۱۵۲۸	۱۳,۷۷۸۰	۰,۵۴۴۲	۲۴۶,۸۸۸۴
σ=100				
Min-Max Kernel K means (β=0)	۳,۱۰۶۵	۱,۸۴۴۰	۰,۵۴۴۳	۱۰۲,۶۷۰۹
Min-Max Kernel K means (β=0.5)	۳,۱۰۶۵	۱,۸۴۴۰	۰,۵۴۴۳	۱۰۲,۶۷۰۹
Min-Max Kernel K means (β=1)	۳,۰۲۳۸	۱,۸۴۲۲	۰,۵۴۴۲	۱۰۲,۳۳۲۸
Kernel K means	۲۱,۱۵۲۸	۱۳,۷۷۸۰	۰,۵۴۴۲	۲۴۶,۸۸۸۴

در دیتاست cancer جدول (۸)، الگوریتم ارائه شده زمانی که σ برابر ۱ است عملکرد خوبی دارد. درمیان تمامی الگوریتم‌ها مقدار SSE و ماکزیمم واریانس و مجموع واریانس داخل خوشه‌ای، برای الگوریتم ارائه شده کمترین مقدار است.

جدول (۸) دیتاست Cancer

	Sum of variance	Max of variance	Silhouette coefficient	SSE
Min-Max K means (β=0)	۲۹۴۱۹۳۸,۳۵۴۵۱۶	۱۹۹۹۱۴۳,۶۸۴۴۱۴	۰,۶۹۸۳۶۳	۸۶۴۰۰۰,۴۳۳۴۱۶
Min-Max K means (β=0.5)	۲۹۴۱۹۳۸,۳۵۴۵۱۶	۱۹۹۹۱۴۳,۶۸۴۴۱۴	۰,۶۹۸۳۶۳	۸۶۴۰۰۰,۴۳۳۴۱۶
Min-Max K means (β=1)	۲۷۷۰۳۳۹,۴۸۳۶۸۷	۱۸۷۹۴۷۲,۳۹۱۹۲۰	۰,۶۹۷۲۶۵	۸۳۵۷۸۲,۵۸۶۲۶۸
Traditional K means	۳۰۷۸۰۰۰	۲۱۹۵۷۰۰	۰,۶۹۷۳	۴۱۹۱۴۰
σ=1				

Min-Max Kernel K means (β=0)	۱۸۴۹۷۰۰	۱۴۷۸۴۰۰	۰,۴۸۰۷	۲۵۸۳۳۰
Min-Max Kernel K means (β=0.5)	۱۸۴۹۷۰۰	۱۴۷۸۴۰۰	۰,۴۸۰۷	۲۵۸۳۳۰
Min-Max Kernel K means (β=1)	۱۸۴۹۷۰۰	۱۴۷۸۴۰۰	۰,۴۸۰۷	۲۵۸۳۳۰
Kernel K means	۱۸۴۹۷۰۰	۱۴۷۸۴۰۰	۰,۴۸۰۷	۲۵۸۳۳۰
σ=50				
Min-Max Kernel K means (β=0)	NAN	NAN	NAN	NAN
Min-Max Kernel K means (β=0.5)	۹۹۹۰۰۰	۷۳۰۳۰۰	۰,۲۲۰۳	۲۲۳۰۹۰
Min-Max Kernel K means (β=1)	۱۲۳۰۱۰۰	۹۶۰۲۲۰	۰,۰۶۸۰	۲۴۳۰۹۰
Kernel K means	۲۱۶۳۲۰۰	۱۷۴۱۴۰۰	۰,۰۶۸۰	۳۵۹۹۵۰
σ=100				
Min-Max Kernel K means (β=0)	۷۶۳۶۰۰	۵۲۲۲۶۰	۰,۳۸۶۶	۱۹۸۰۶۰
Min-Max Kernel K means (β=0.5)	۷۷۱۳۰۰	۵۲۷۸۷۰	۰,۳۸۲۱	۱۹۸۷۷۰
Min-Max Kernel K means (β=1)	۱۰۸۵۶۰۰	۸۰۵۳۲۰	۰,۱۵۶۵	۲۳۲۸۳۰
Kernel K means	۲۰۰۳۰۰۰	۱۵۵۹۷۰۰	۰,۱۵۶۵	۳۵۵۴۵۰

در دیتاست CMC با توجه جدول (۹) به ازای σ برابر ۱۰۰ و بتا برابر ۱ بهترین کارایی را دارد.

جدول (۹) دیتاست CMC

	Sum of variance	Max of variance	Silhouette coefficient	SSE
Min-Max K means (β=0)	۷۹۹,۴۹۱۲۸۴	۳۸۸,۹۶۳۲۹۸	۰,۴۳۸۵۷۱	۳۸۲۴۹,۳۳۶۴۴۵
Min-Max K means (β=0.5)	۷۹۹,۴۹۱۲۸۴	۳۸۸,۹۶۳۲۹۸	۰,۴۳۸۵۷۱	۳۸۲۴۹,۳۳۶۴۴۵
Min-Max K means (β=1)	۷۹۹,۴۹۱۲۸۴	۳۸۸,۹۶۳۲۹۸	۰,۴۳۸۵۷۱	۳۸۲۴۹,۳۳۶۴۴۵
Traditional K means	۱۱۷۴,۶	۵۴۹,۵۲۶۵	۰,۴۴۲۸	۱۶۱۶۶
σ=1				
Min-Max Kernel K means (β=0)	NAN	NAN	NAN	NAN
Min-Max Kernel K means (β=0.5)	NAN	NAN	NAN	NAN
Min-Max Kernel K means (β=1)	۵۳۳,۲۴۳۵	۳۶۲,۶۹۵۷	-۰,۰۲۷۲	۹۱۸۲,۱
Kernel K means	۱۱۳۷,۲	۷۳۳,۰۴۳۸	-۰,۰۲۷۲	۱۳۰۰۴
σ=50				
Min-Max Kernel K means (β=0)	۹۸,۶۹۵۸	۴۷,۴۱۴۲	۰,۴۳۹۷	۵۵۴۳,۶
Min-Max Kernel K means (β=0.5)	۹۸,۶۸۷۵	۴۷,۴۲۲۶	۰,۴۳۹۸	۵۵۴۳,۶
Min-Max Kernel K means (β=1)	۹۸,۷۵۳۷	۴۷,۴۸۱۴	۰,۴۴۰۱	۵۵۴۳,۷
Kernel K means	۲۹۱,۲۷۶۹	۱۳۷,۲۹۷۷	۰,۴۴۰۱	۸۶۶۴,۱
σ=100				
Min-Max Kernel K means (β=0)	۹۸,۶۸۷۵	۴۷,۴۲۲۶	۰,۴۳۹۸	۵۵۴۳,۶
Min-Max Kernel K means (β=0.5)	۹۸,۶۸۷۵	۴۷,۴۲۲۶	۰,۴۳۹۸	۵۵۴۳,۶
Min-Max Kernel K means (β=1)	۹۸,۷۵۳۷	۴۷,۴۸۱۴	۰,۴۴۰۱	۵۵۴۳,۷
Kernel K means	۲۹۱,۲۷۶۹	۱۳۷,۲۹۷۷	۰,۴۴۰۱	۸۶۶۴,۱

برای دیتاست vowel جدول (۱۰)، الگوریتم ارائه شده از نظر معیار silhouette کارایی پایین‌تری نسبت به الگوریتم ارائه شده در [15] و K means داشته است.

جدول (۱۰) دیتاست vowel

	Sum of variance	Max of variance	Silhouette coefficient	SSE
Min-Max K means (β=0)	۱۲۱۹۴۹۵۵,۷۶۹۶۶۵	۴۲۷۹۳۷۱,۴۷۹۹۶۵	۰,۳۶۲۵۳۶	۲۰۰۹۷۵۷,۳۳۰۳۱۷
Min-Max K means (β=0.5)	۱۲۴۱۹۲۱۳,۴۲۶۹۹۴	۴۶۲۵۴۸۵,۶۴۰۱۸۴	۰,۳۵۹۸۳۳	۱۹۹۹۱۳۷,۹۶۱۹۱۱
Min-Max K means (β=1)	۱۲۲۲۲۶۳۹,۹۵۴۳۸۰	۳۹۰۲۰۵۶,۸۰۹۳۶۵	۰,۳۶۳۹۸۱	۲۰۲۲۳۲۲,۸۱۷۲۴۲
Traditional K means	۱۷۷۴۸۰۰۰	۴۵۷۴۶۰۰	۰,۳۸۱۵	۶۲۵۹۷۰
σ=1				
Min-Max Kernel K means (β=0)	NAN	NAN	NAN	NAN
Min-Max Kernel K means (β=0.5)	۵۸۲۴۷۰۰	۳۳۷۶۴۰۰	۰,۱۷۲۶	۲۰۶۹۷۰
Min-Max Kernel K means (β=1)	۵۳۵۵۶۰۰	۳۰۲۸۳۰۰	۰,۱۷۶۸	۲۱۴۴۶۰
Kernel K means	۸۲۶۴۹۰۰	۵۰۸۴۰۰۰	۰,۱۷۶۸	۲۵۶۹۷۰
σ=50				
Min-Max Kernel K means (β=0)	NAN	NAN	NAN	NAN
Min-Max Kernel K means (β=0.5)	NAN	NAN	NAN	NAN
Min-Max Kernel K means (β=1)	۸۲۸۹۵۰۰	۵۵۷۰۲۰۰	۰,۰۲۰۵	۲۷۱۲۳۰
Kernel K means	۱۳۹۸۱۰۰۰	۱۰۰۴۵۰۰۰	۰,۰۲۰۵	۳۴۳۹۲۰
σ=100				
Min-Max Kernel K means (β=0)	۱۲۱۰۵۰۰	۳۶۶,۶۰۰	۰,۳۱۵۴	۱۶۵۶۶۰
Min-Max Kernel K means (β=0.5)	۱۴۳۸۵۰۰	۴۲۲۲,۲۰۰	۰,۲۹۷۶	۱۷۴۷۱۰
Min-Max Kernel K means (β=1)	۳۵۳۷۲۰۰	۲۱۲۱۹۰۰	۰,۲۶۶۳	۱۸۱۹۷۰
Kernel K means	۷۲۳۶۷۰۰	۴۵۰۸۴۰۰	۰,۲۶۶۳	۲۶۰۸۹۰

برای دیتاست thyroid با توجه به جدول (۱۱)، الگوریتم ارائه شده از نظر معیار silhouette شبیه به الگوریتم Min-Max K means عمل می‌کند. به‌ازای سایرمعیارها الگوریتم ارائه شده بهترین کارایی را دارد.

جدول (۱۱) دیتاست Thyroid

	Sum of variance	Max of variance	Silhouette coefficient	SSE
Min-Max K means (β=0)	۳۹۲۸,۷۶۵۳۰۵	۲۱۵۱,۳۲۶۲۸۳	۰,۴۰۲۰۷۰	۱۱۱۵۳,۷۶۶۲۱۴
Min-Max K means (β=0.5)	۳۹۹۱,۷۱۷۳۶۷	۲۲۴۴,۳۳۳۴۸۲	۰,۴۰۳۲۶۰	۱۱۰۸۱,۰۲۱۷۳۶
Min-Max K means (β=1)	۳۹۲۴,۸۵۳۳۰۳	۲۰۹۵,۷۵۱۵۳۵	۰,۴۴۲۷۷۹	۱۱۶۲۱,۱۲۱۳۱۲
Traditional K means	۴۰۸۴,۵	۲۳۷۷,۱	۰,۴۶۶۸	۳۱۰۳,۹
σ=1				
Min-Max Kernel K means (β=0)	۱۷۴۱,۴	۱۱۲۸,۴	۰,۲۱۱۴	۲۲۳۹,۷
Min-Max Kernel K means (β=0.5)	۱۷۴۱,۴	۱۱۲۸,۴	۰,۲۱۱۴	۲۲۳۹,۷
Min-Max Kernel K means (β=1)	۱۷۴۱,۳	۱۱۲۷,۹	۰,۲۱۱۲	۲۲۴۰
Kernel K means	۱۸۰۵,۳	۱۱۵۷,۸	۰,۲۱۱۲	۲۲۶۱,۲
σ=50				
Min-Max Kernel K means (β=0)	۱۴۰۹,۸	۸۸۰,۹۵۵۳	۰,۴۳۸۳	۱۹۸۳
Min-Max Kernel K means (β=0.5)	۱۴۲۴	۸۸۹,۰۸۶۸	۰,۴۳۹۰	۱۹۸۳,۹
Min-Max Kernel K means (β=1)	۱۴۰۹,۸	۸۹۷,۱۲۹۶	۰,۴۳۸۱	۱۹۸۲,۳
Kernel K means	۲۵۹۸,۲	۱۶۶۴,۴	۰,۴۳۸۱	۲۵۰۵

σ=100				
Min-Max Kernel K means (β=0)	۱۴۰۸,۲	۸۷۶,۴۴۲۴	۰,۴۴۱۷	۱۹۸۸,۲
Min-Max Kernel K means (β=0.5)	۱۴۰۷,۱	۸۸۳,۵۶۸۶	۰,۴۴۲۸	۱۹۹۱,۱
Min-Max Kernel K means (β=1)	۱۴۸۲,۲	۹۶۰,۴۰۳۳	۰,۴۴۱۷	۱۹۸۸,۲
Kernel K means	۲۷۲۹,۷	۱۷۷۷,۳	۰,۴۴۱۷	۲۵۰۷,۶

با مشاهده همه‌ی دیتاست‌ها می‌توان اینگونه نتیجه می‌گیریم که به ازای تمامی مقادیر بتا (تاثیر گزاری حافظه)، مقدار ماکزیمم واریانس یک مقدار کوچک است. این به طور واضح نشان می‌دهد که الگوریتم توانسته ماکزیمم واریانس خوشه‌ها را مینیمم کند یعنی کیفیت خوشه بندی را بهبود بخشد. الگوریتم از نظر زمان اجرا در جایگاه سوم قرار دارد دلیل این امر پیچیدگی محاسباتی است که در این الگوریتم وجود دارد و تعداد تکرار زیادی است که الگوریتم باید همگرا شود. از لحاظ سرعت، K means در جایگاه اول قرار دارد ولی از نظر دقت در خوشه بندی عملکردی خوبی ندارد و در بیشتر مواقع از مقدار SSE بسیار بالایی برخوردار است. با بررسی مقدار تاثیر گزاری حافظه (بتا) به این نتیجه رسیدم که با افزایش این مقدار کاهش زیادی در ماکزیمم واریانس امکان پذیر است و همچنین از به‌وجود آمدن خوشه‌های خالی و یا با یک مقدار خوداری می‌شود. الگوریتم حتی زمانی که مقدار بتا برابر صفر است، عملکرد بهتری نسبت به بقیه دارد. با بررسی مقدار σ و تاثیر آن بر روی خوشه بندی داده‌ها، با توجه به آزمایشات انجام شده، به این نتیجه رسیدیم که با افزایش مقدار σ، معیار silhouette افزایش می‌یابد و به بهترین مقدار ممکن در مقایسه با سایر الگوریتم‌ها می‌توان دست یافت. ولی مقدار واریانس داخل خوشه‌ای نامتعادل می‌شود که این نشان دهنده کیفیت بد خوشه بندی است.

در نهایت می‌توان گفت الگوریتم ارائه شده در بسیاری از حالات بهتر از سایر الگوریتم ها عملکرده است و در مقابله به وجود آمدن خوشه ای با واریانس بالا جلوگیری کرده است.

۵- نتیجه گیری

ما الگوریتم Min-Max Kernel K means ارائه کردیم که مشکل مربوط به نقاط اولیه و نامناسب بودن برای داده‌های غیرخطی و واگرا، در K means را حل می‌کند. وزن‌هایی به خوشه‌ها با توجه به واریانس آنها اختصاص دادیم. در واقع نسخه K means وزن دار را بهبود دادیم و از بوجود آمدن خوشه‌هایی با واریانس نامتعدل و بالا جلوگیری کردیم. توان p را که به وسیله کاربر مقداردهی می‌شود، برای کنترل سختگیرانه الگوریتمان در رویارویی با خوشه‌هایی با واریانس بزرگ، تخصیص دادیم. با تنبیه خوشه‌هایی با واریانس بزرگتر، به مشکل مربوط به مقدار دهیه اولیه غلبه کردیم و باعث بوجود آمدن خوشه‌هایی با کیفیت شدیم. همچنین خوشه‌ها از نظر واریانس متعادل شدند. آموزش به صورت تکراری انجام شد که در آن وزنها در مرحله ماکزیمم سازی آپدیت شدند. همچنین ما یک بهبودی برای پیدا کردن مقدار p ارائه کردیم که به عنوان یک تسهیل کننده الگوریتم، بهترین خوشه بندی را با توجه به مقدار p ارائه می‌دهد. برای ارزیابی الگوریتممان آن را بر روی چندین دیتاست امتحان کرده‌ایم. نتایجی که حاصل شده نشان داد الگوریتم ارائه شده به انتخاب نقاط اولیه مناسب حساس نیست و حتی با انتخاب نقاط اولیه نامناسب نیز کارایی بهتری نسبت به دیگر روش‌های مقایسه شده دارد. همچنین روش ارائه شده را با سه الگوریتم K means، Kernel K means و Min-Max K means مقیاسه کردیم به این نتیجه رسیدیم که الگوریتم ارائه شده عملکرد بهتری در شرایط یکسان با آن‌ها دارد.

به عنوان یک کار آینده قصد داریم الگوریتم فازی سی مینز را همانند الگوریتم ارائه شده در این مقاله بهبود دهیم و راه حلی برای پیدا کردن مقدار بتا بدست بیاوریم.

مراجع

1. Filippone, M., et al., *A survey of kernel and spectral methods for clustering*. Pattern recognition, 2008. 41(1): p. 176-190.
2. Xu, R. and D. Wunsch, *Survey of clustering algorithms*. IEEE Transactions on neural networks, 2005. 16(3): p. 645-678.
3. Muller, K.-R., et al., *An introduction to kernel-based learning algorithms*, in *Handbook of Neural Network Signal Processing*. 2001, CRC Press.
4. Girolami, M., *Mercer kernel-based clustering in feature space*. IEEE Transactions on Neural Networks, 2002. 13(3): p. 780-784.
5. Arthur, D. and S. Vassilvitskii. *k-means++: The advantages of careful seeding*. in *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*. 2007. Society for Industrial and Applied Mathematics.
6. Wang, C.-D., J.-H. Lai, and J.-Y. Zhu. *A conscience on-line learning approach for kernel-based clustering*. in *Data Mining (ICDM), 2010 IEEE 10th International Conference on*. 2010. IEEE.
7. Banerjee, A. and J. Ghosh, *Frequency-sensitive competitive learning for scalable balanced clustering on high-dimensional hyperspheres*. IEEE

Transactions on Neural Networks, 2004. 15(3): p. 702-719.
8. Bradley, P.S. and U.M. Fayyad. *Refining Initial Points for K-Means Clustering*. in *ICML*. 1998. Citeseer.
9. Su, T. and J.G. Dy, *In search of deterministic methods for initializing K-means and Gaussian mixture clustering*. Intelligent Data Analysis, 2007. 11(4): p. 319-338.
10. Likas, A., N. Vlassis, and J.J. Verbeek, *The global k-means clustering algorithm.* Pattern recognition, 2003. 36(2): p. 451-461.
11. Bagirov, A.M., *Modified global k-means algorithm for minimum sum-of-squares clustering problems.* Pattern Recognition, 2008. 41(10): p. 3192-3199.
12. Bagirov, A.M., J. Ugon, and D. Webb, *Fast modified global k-means algorithm for incremental cluster construction*. Pattern recognition, 2011. 44(4): p. 866-876.
13. Tzortzis, G. and A. Likas. *The global kernel k-means clustering algorithm*. in *Neural Networks, 2008. IJCNN 2008.(IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on*. 2008. IEEE.
14. Tzortzis, G.F. and A.C. Likas, *The Global Kernel $ k $-Means Algorithm for Clustering in Feature Space.* IEEE Transactions on Neural Networks, 2009. 20(7): p. 1181-1194.
15. Tzortzis, G. and A. Likas, *The MinMax k-Means clustering algorithm.* Pattern Recognition, 2014. 47(7): p. 2505-2516.
16. Dhillon, I.S., Y. Guan, and B. Kulis, *A unified view of kernel k-means, spectral clustering and graph cuts*. 2004: Citeseer.
17. Schölkopf, B., A. Smola, and K.-R. Müller, *Nonlinear component analysis as a kernel eigenvalue problem.* Neural computation, 1998. 10(5): p. 1299-1319.
18. Pena, J.M., J.A. Lozano, and P. Larranaga, *An empirical comparison of four initialization methods for the k-means algorithm.* Pattern recognition letters, 1999. 20(10): p. 1027-1040.
19. Celebi, M.E., H.A. Kingravi, and P.A. Vela, *A comparative study of efficient initialization methods for the k-means clustering algorithm.* Expert Systems with Applications, 2013. 40(1): p. 200-210.
20. www.UCI.com

زیرنویس‌ها

۱ Isotropic
۲ Non-isotropic
۳ Linear separable
۴ Non- Linear separable
۵ Input space
۶ Feature space
۷ Nonlinear transformation
۸ Dot product
۹ Kernel induced feature space
۱۰ Objective function
۱۱ Minimize the maximum intra-cluster variance
۱۲ Memory effect