به نام خدا

تمرین درس شناسایی آماری الگو –سری چهارم

فردین آیار

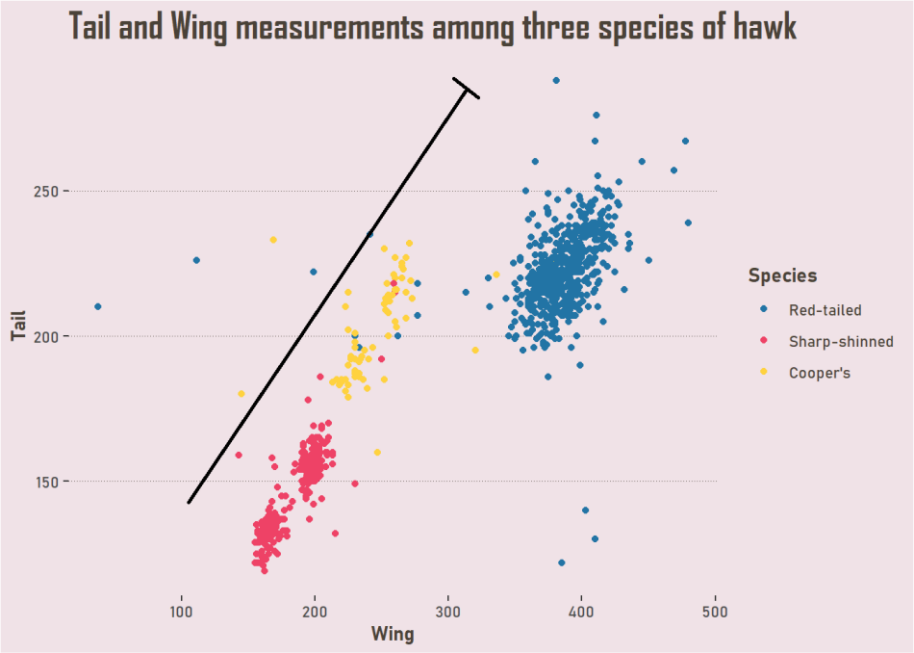
شماره دانشجویی: ۹۹۱۳۱۰۴۰

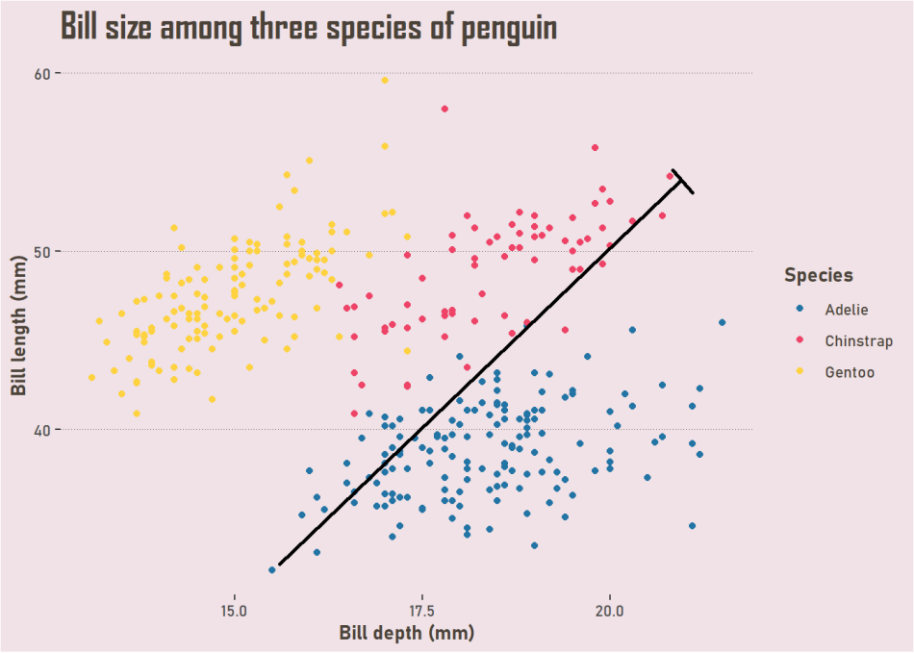
استاد: دکتر رحمتی

دانشکده کامپیوتر– زمستان۹۹

۱)

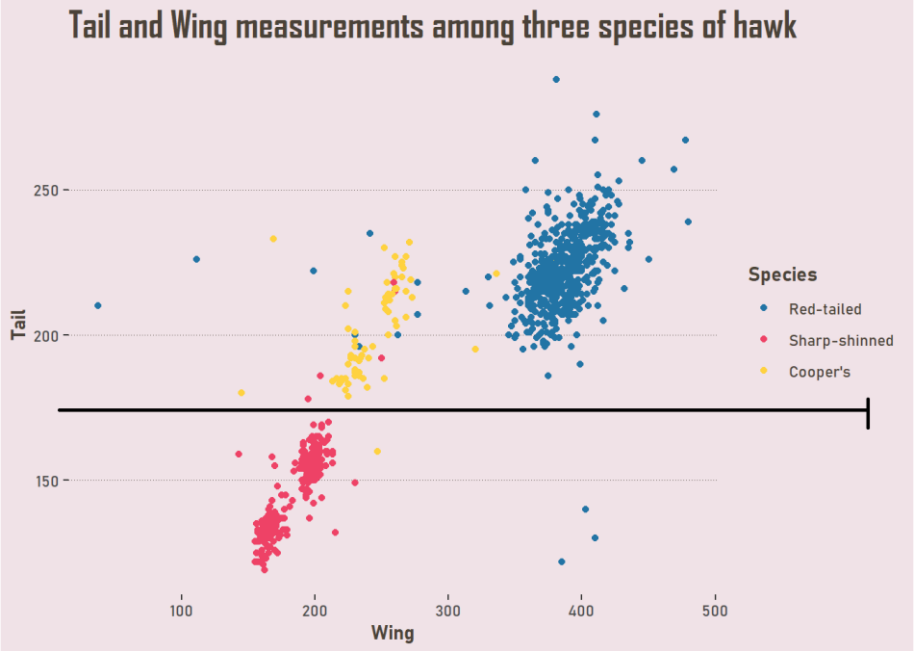
a)

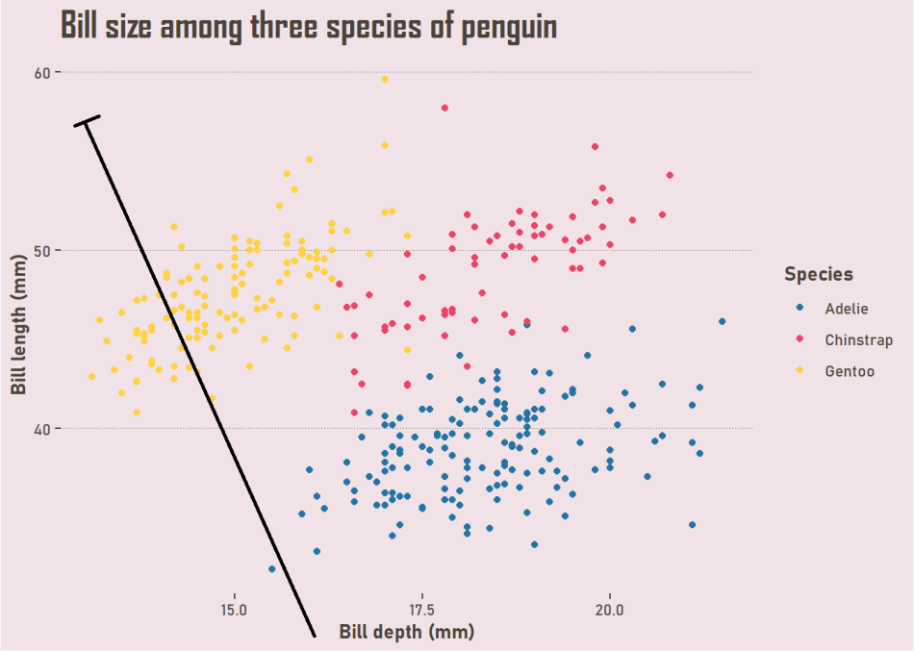




b) در اعمال LDA از آنجا که هم پراکندگی و هم فاصله میانگین‌ها مهم است، تشخیص چشمی آن اندکی دشوارتر است.

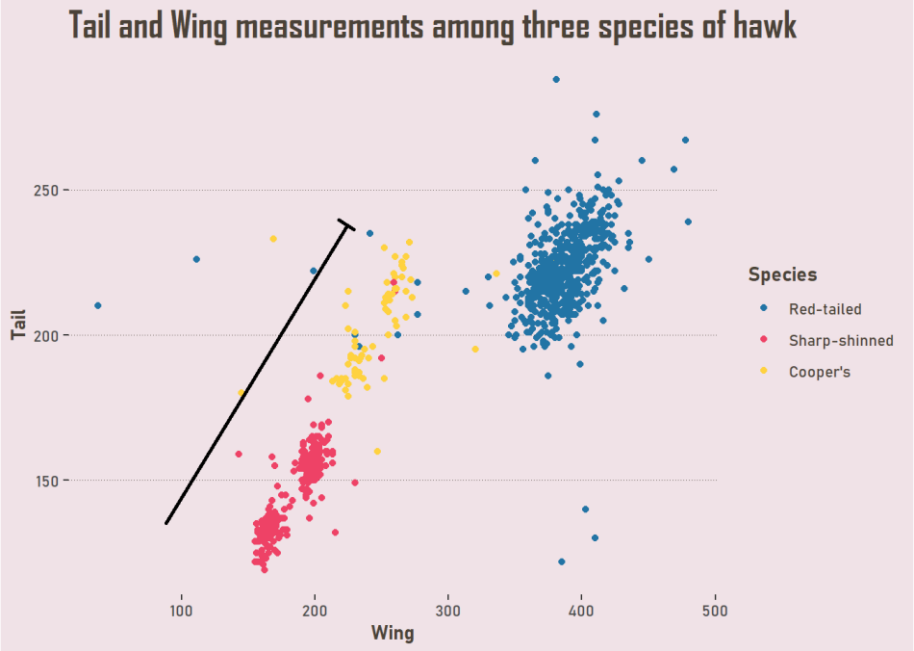
3

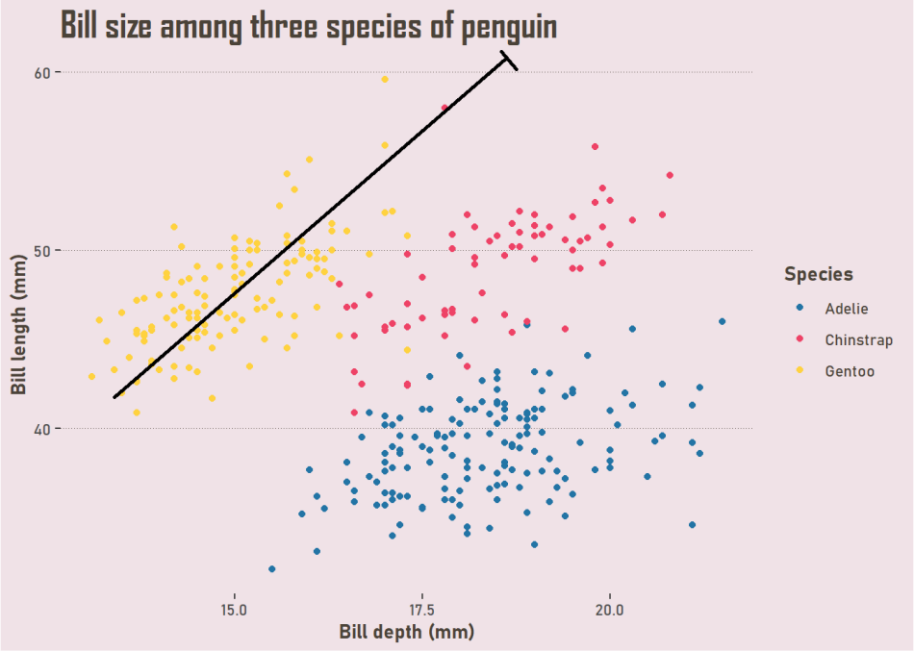

Tail and Wing measurements among three species of hawk
Tail
250
200
150
100
200
300
400
500
Wing
Species
Red-tailed
Sharp-shinned
Cooper's


Bill size among three species of penguin
Bill length (mm)
60
50
40
15.0
17.5
20.0
Bill depth (mm)
Species
Adelie
Chinstrap
Gentoo

(c

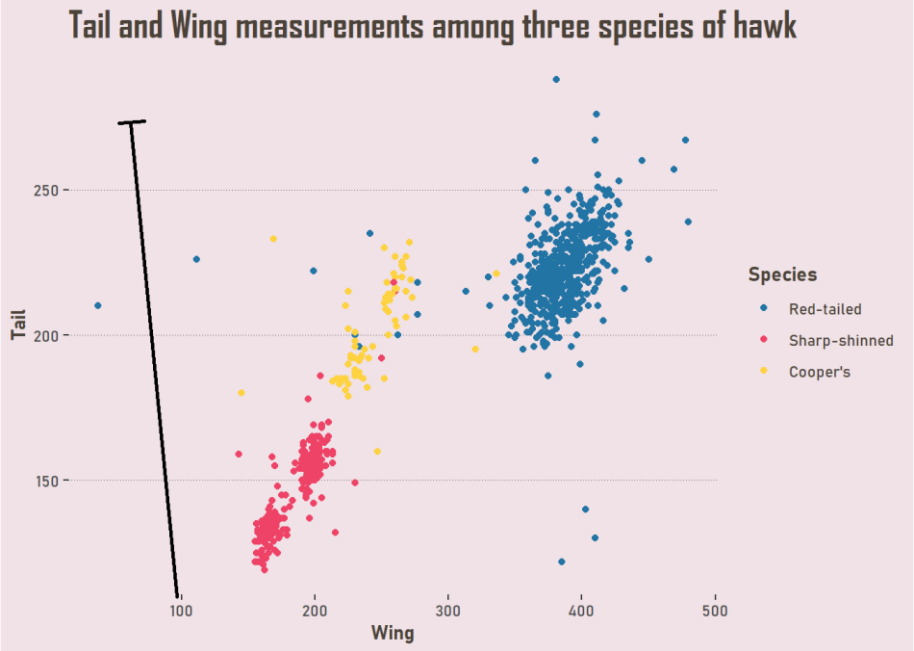
4

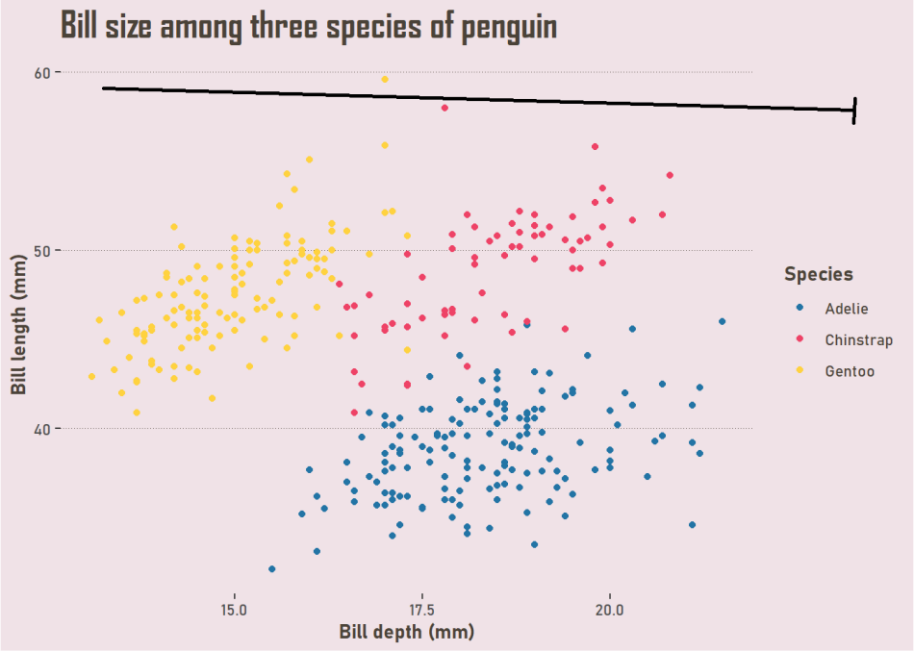

Tail and Wing measurements among three species of hawk
Tail
250
200
150
100
200
300
400
500
Wing
Species
Red-tailed
Sharp-shinned
Cooper's


Bill size among three species of penguin
Bill length (mm)
60
50
40
15.0
17.5
20.0
Bill depth (mm)
Species
Adelie
Chinstrap
Gentoo

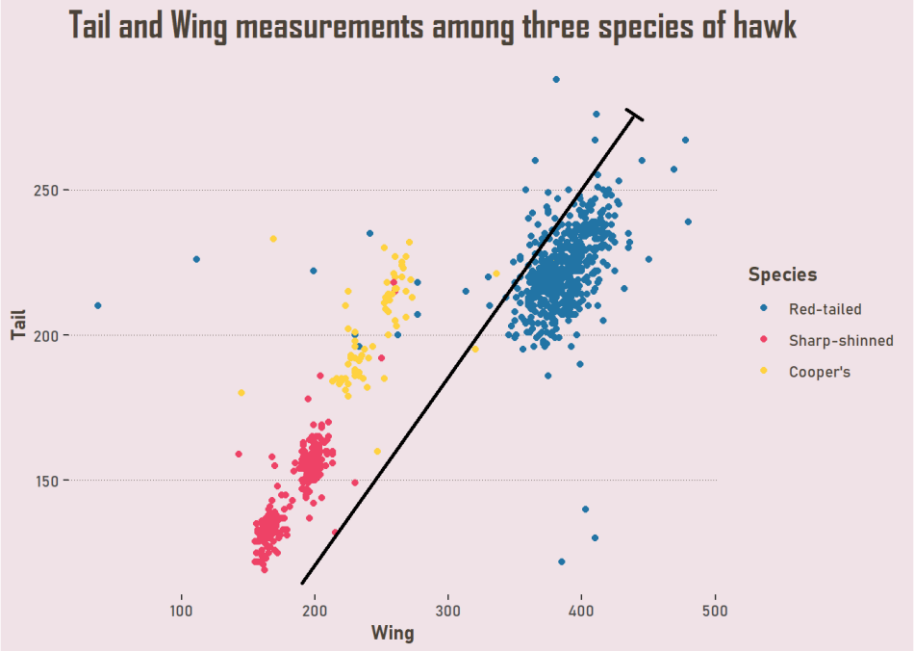
(d

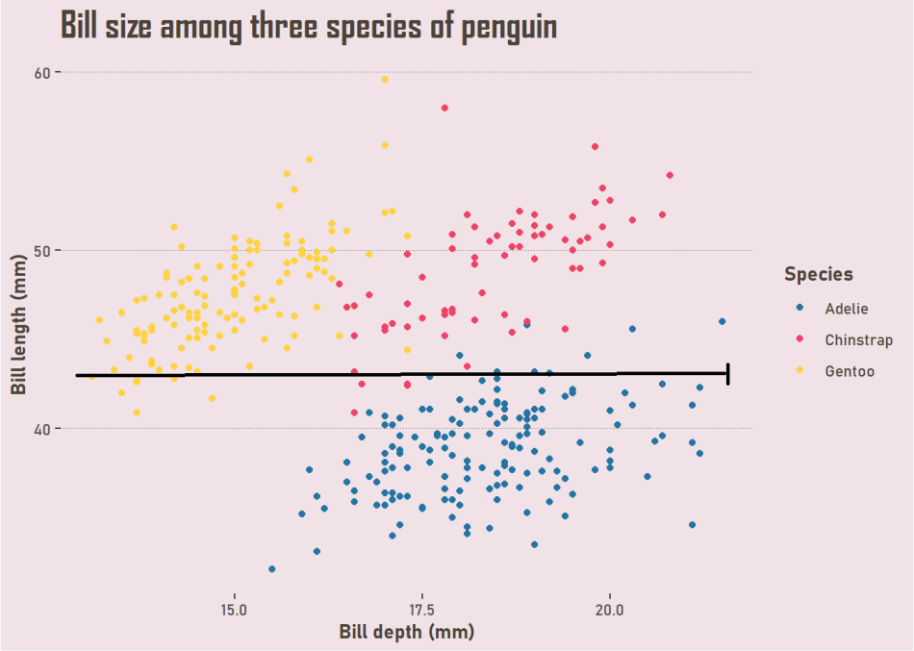
5


Tail and Wing measurements among three species of hawk
Tail
250
200
150
100
200
300
400
500
Wing
Species
Red-tailed
Sharp-shinned
Cooper's

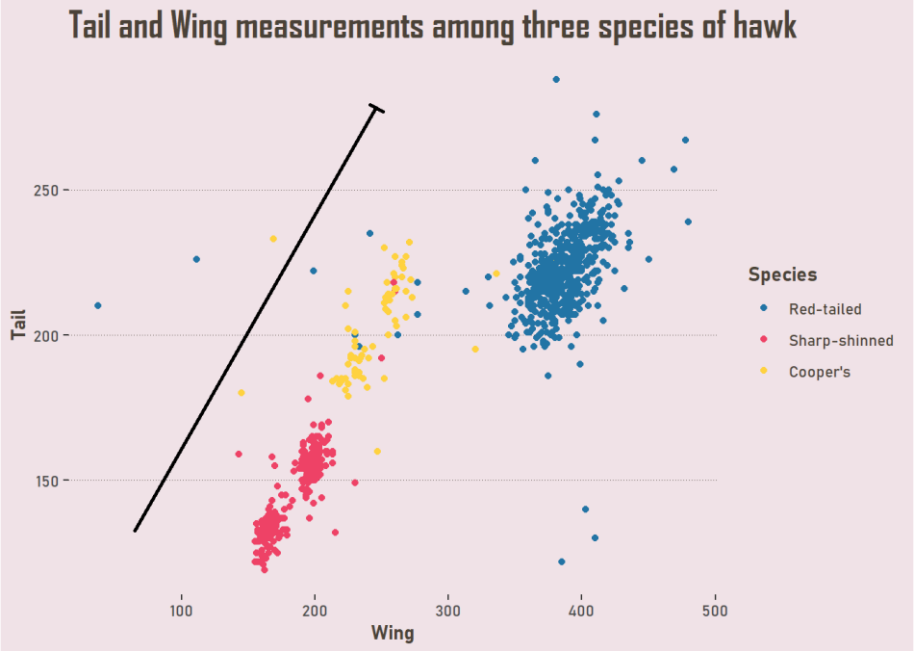

Bill size among three species of penguin
Bill length (mm)
60
50
40
15.0
17.5
20.0
Bill depth (mm)
Species
Adelie
Chinstrap
Gentoo

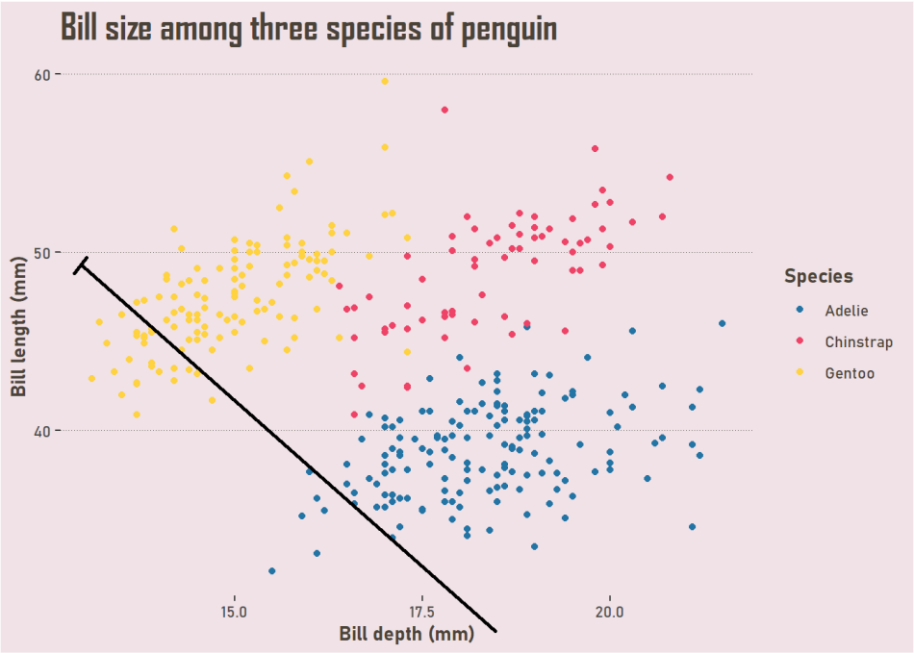
(e


Tail and Wing measurements among three species of hawk
Tail
250
200
150
100
200
300
400
500
Wing
Species
Red-tailed
Sharp-shinned
Cooper's


Bill size among three species of penguin
Bill length (mm)
60
50
40
15.0
17.5
20.0
Bill depth (mm)
Species
Adelie
Chinstrap
Gentoo

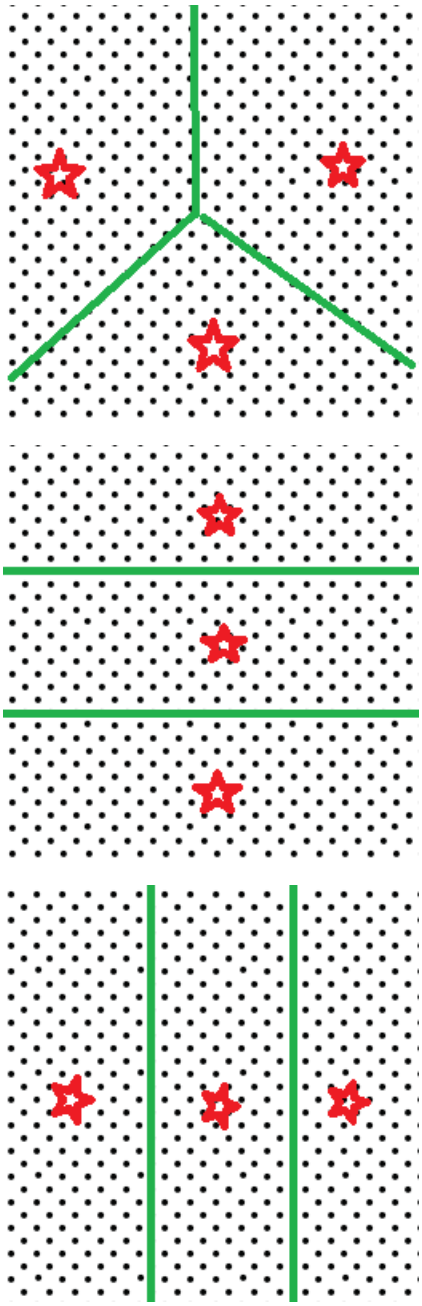
(f



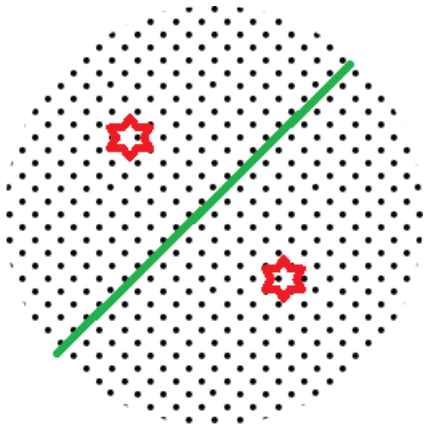


۲)

a) سه شکل زیر نمونه‌ای از مرزهای کلاسترها را به همراه مرکز آنها، نشان می‌دهد. شکل اول بهینه سراسری و دو شکل بعدی محلی هستند.

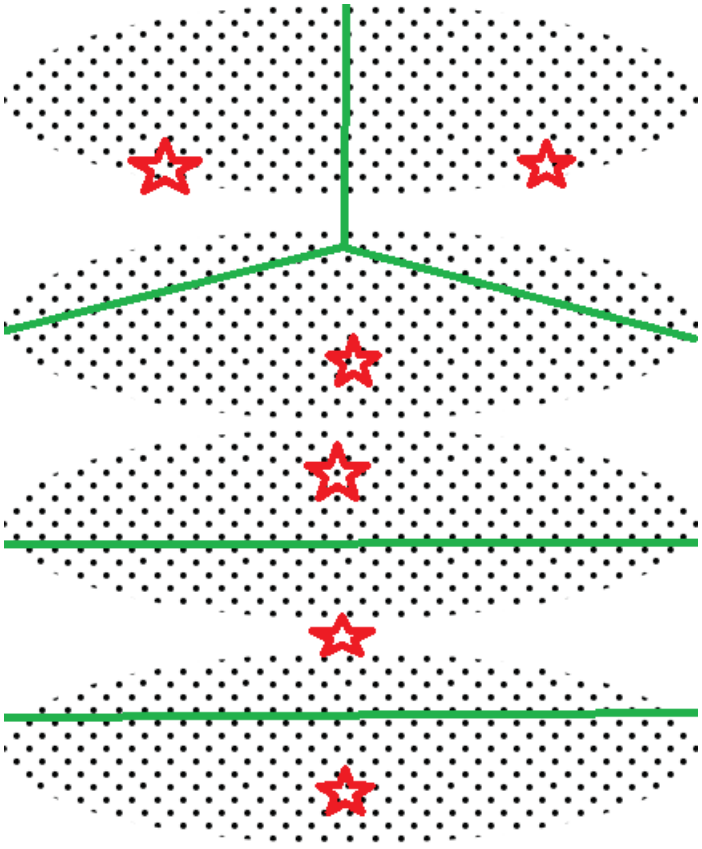


b) برای این شکل مرز کلاسترها، قطرهای دایره هستند؛ بنابراین بی‌نهایت خروجی مختلف با توجه به مرکزهای اولیه می‌تواند وجود داشته باشد. یکی از جواب های ممکن در شکل زیر ارائه شده است.

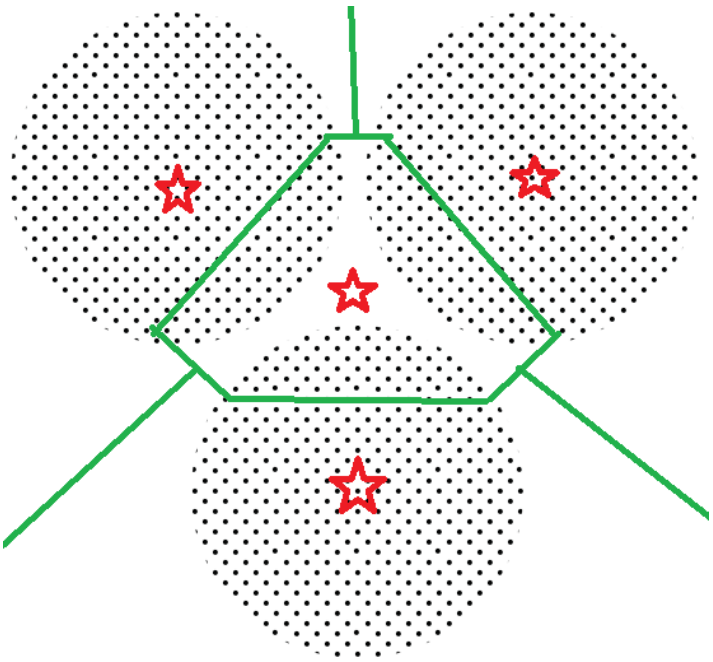


لازم به ذکر است مرکز کلاسترها دقیقا در مرکز سطح نیم دایره‌ها قرار می‌گیرند.(اگرچه در شکل ارائه شده این نکته ممکن است مشهود نباشد)

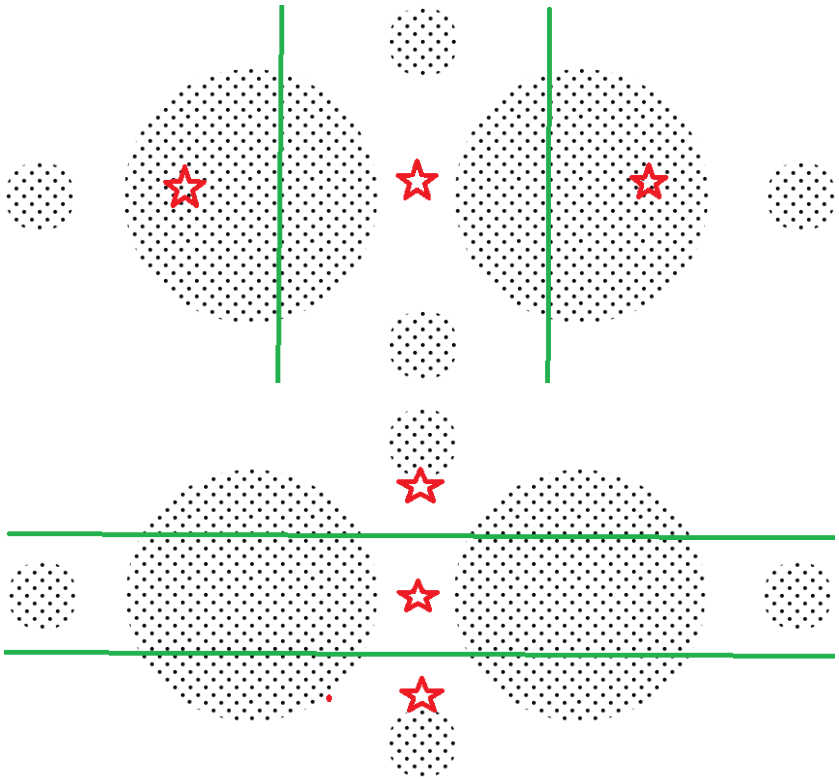
c) به نظر می‌رسد واریانس داده‌ها در تصویر اول کمتر باشد، بنابراین این تصویر بهینه‌سراسری باشد:



d) شکل نهایی به صورت زیر می‌باشد:



e) تصویر اول بهینه سراسری و تصویر دوم بهینه محلی است:



f) شکل ۲- تصویر سمت راست

g) الگوریتم‌های خوشبه‌بندی در تشخیص الگوهایی که از عدم وجود نقاط به وجود می‌آید(مانند الگوی در و پنجره در شکل ۱ قسمت f) ناکارآمد هستند.

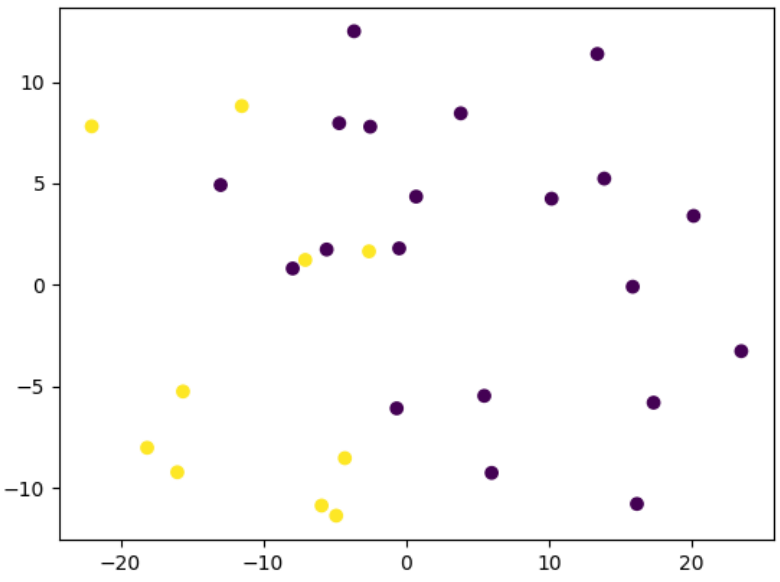
۳)

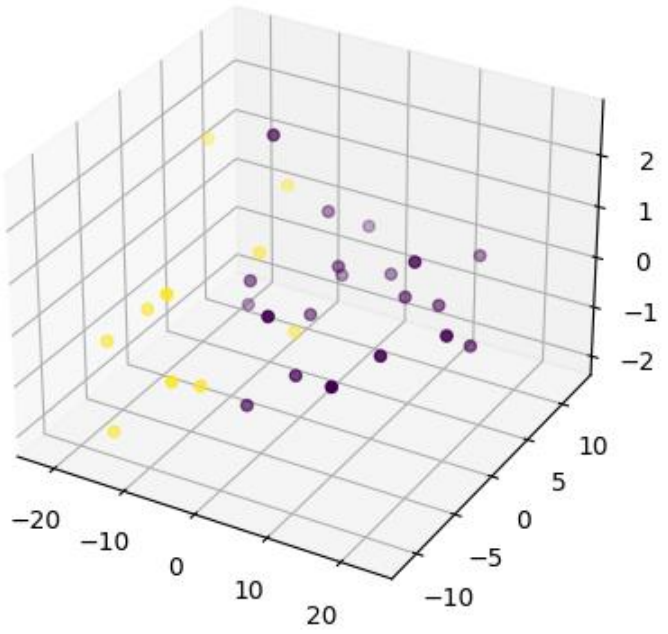
a) کد مربوط به این قسمت در فایل a.py قراردارد. تابع پیاده سازی شده، ضمن محاسبه n مولفه اصلی، تمام مقادیرویژه مربوط به داده‌ها را به صورت مرتب چاپ می‌کند. با توجه به این تابع، مقادیرویژه داده‌ها به این صورت می‌باشد:(اعداد تقریبی هستند)

[4222, 1542, 26, 8, 1, 0.3]

با توجه به مقادیر فوق، سه و حتی دو مولفه اصلی، برای پوشش دادن بزرگترین واریانس‌های داده‌ها کافی به نظر می‌رسند.

c) کد مربوط به این قسمت در فایل c.py قراردارد. نقاط بنفش مربوط به رستوران‌های ۱ تا ۴ (ناپل) و نقاط زردرنگ مربوط به رستوران‌های ۵ و ۶ (سایر رستوران‌ها) می‌باشند.



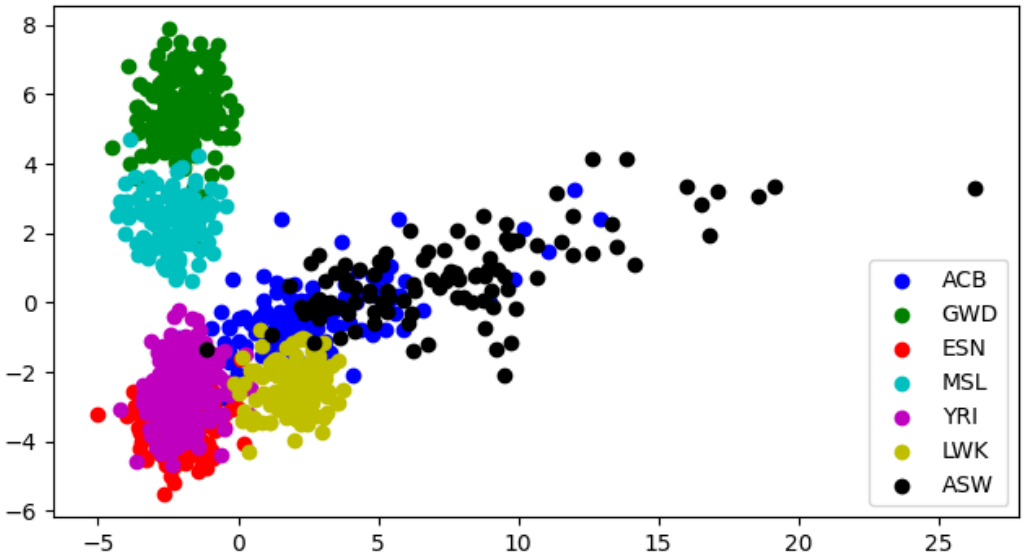


همانطور که مشاهده می‌شود، دو و سه مولفه‌اصلی داده‌ها برای جداکردن داده‌ها، تا حد زیادی مناسب است.

۴)

a) بعد از اعمال PCA، بُعد داده ها برابر با تعداد مولفه‌های‌اصلی انتخاب شده‌است. به عنوان مثال برای قسمت b، بعد از اعمال PCA داده‌ها دوبعدی می‌شوند.

b) کد مربوط به این بخش در فایل b.py قرار دارد. لازم به ذکر است الگوریتم PCA کتابخانه sklearn به صورت پیش فرض داده‌ها را به میانگین صفر منتقل می‌کند.بنابراین لزومی به انتقال دستی آنها نیست. داده‌ها را بعد از انتقال به فضای جدید رسم می‌کنیم:



c) برای تحلیل بهتر، ابتدا جغرافیای هر جمعیت را روی نقشه مشخص می‌کنیم:

باتوجه به شکل فوق و نمودار خروجی بخش b، می‌توان دریافت که این دو مولفه‌اصلی، جنبه جغرافیایی ژنتیک این افراد را نشان می‌دهد. به طور خلاصه، موارد زیر ادعای فوق را تایید می‌کنند:

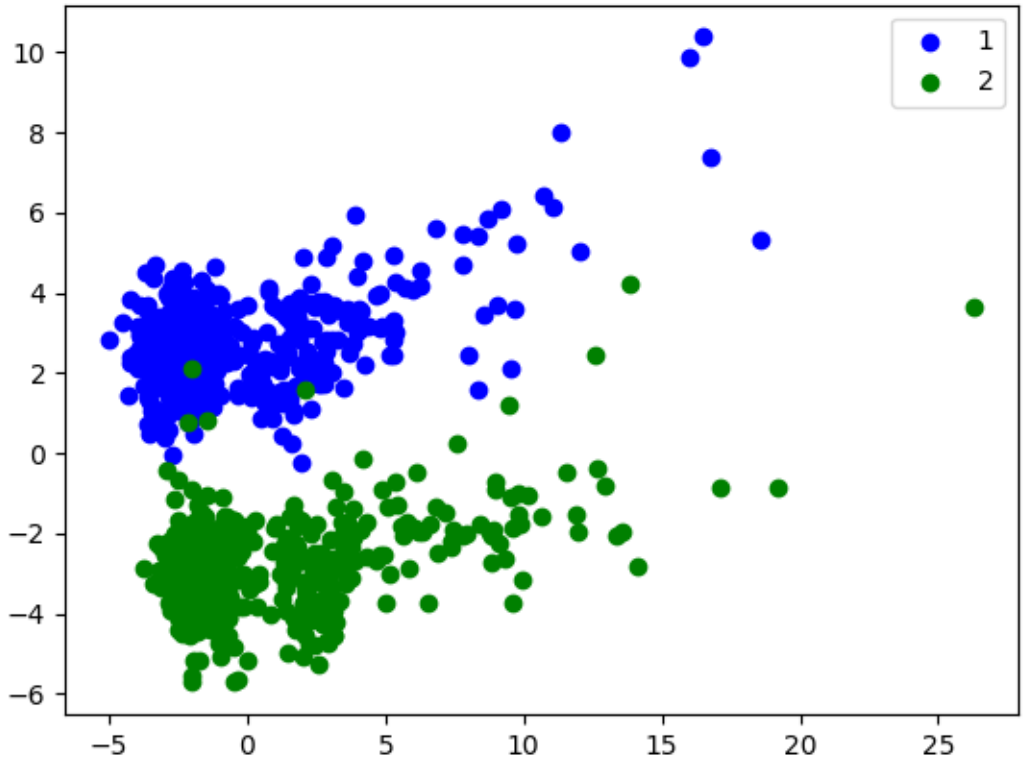
۱) کد ASW که مربوط به آفریقایی‌تباران ساکن شمال‌شرق‌آمریکا می‌باشند، دارای پراکندگی بسیار زیاد روی نمودار بخش b هستند. دلیل این امر احتمالا این است که آمریکا دارای تنوع نژادی بسیار بالایی است و ژنتیک این افراد در طول تاریخ با هم ترکیب شده‌است.

۲) در نمودار، دو کد YRI و ESN تقریبا برهم منطبق هستند. با توجه به نقشه فوق، این دو گروه از نظر جغرافیایی نیز بسیار به هم نزدیک هستند.

۳) گروه ACB دارای تنوع ژنتیکی بالایی است و همچنین طبق نمودار به سه گروه ESN، YRI و LWK نزدیک است. به لحاظ تاریخی، ریشه این گروه که آفریقایی‌تباران ساکن بارباروس هستند، به برده‌هایی برمی‌گردد که اکثر آن‌ها از کشورهایی مانند کنیا(LWK) و نیجریه(ESN و YRI) از طریق خلیج گینه به بارباروس منتقل شده‌اند.

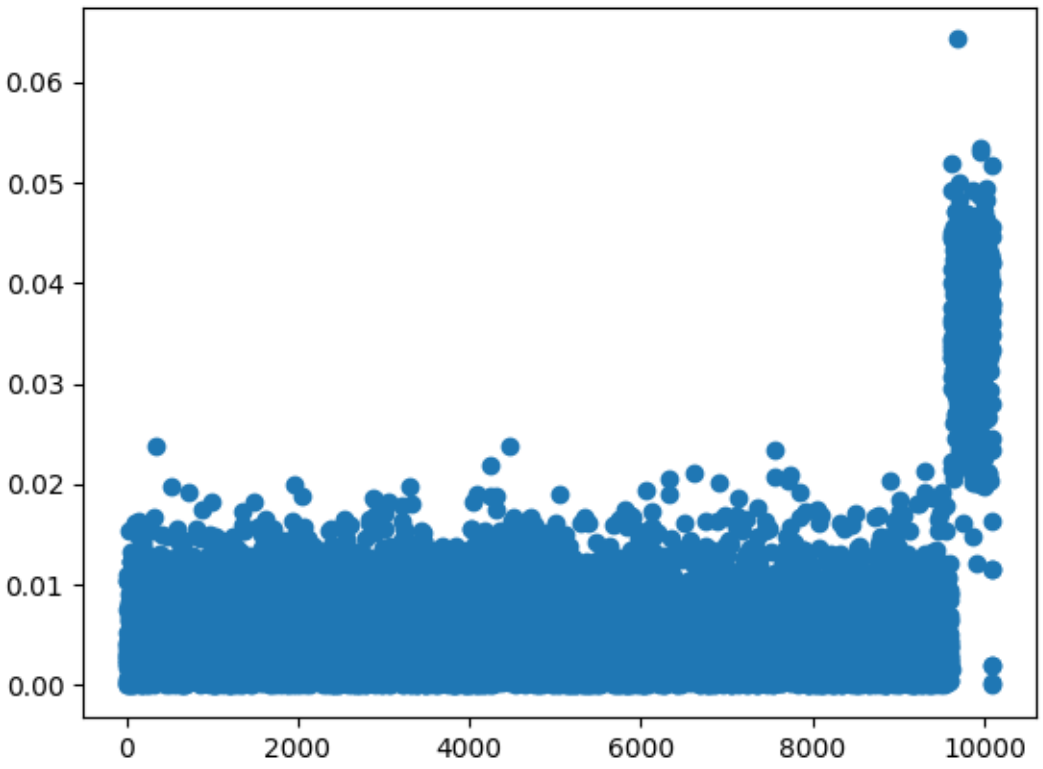
۴) دو گروه GWD و MSL، از سایر گروه‌ها فاصله دارند و چون مربوط به دو کشور مختلف نزدیک به هم هستند، ضمن نزدیک بودن به یکدیگر، کاملا جداپذیر هستند.

d) کد مربوط به این بخش در فایل d.py قرار دارد. خروجی این دو مولفه، افراد را براساس جنسیت به دو کلاستر مختلف تقسیم می‌کنند. خروجی به این صورت است:



e) با توجه به نمودار، مولفه سوم اطلاعات مربوط به جنسیت افراد را شامل می‌شود. با توجه به شکل فوق آستانه Y=0 ، برای تفکیک جنسیت افراد کافی به نظر می‌رسد؛ بنابراین مولفه سوم به تنهایی برای تفکیک جنسیت افراد کافی است.

f) کد مربوط به این بخش در فایل f.py قرار دارد. نمودار خواسته شده در شکل زیر ارائه شده است:



همانطور که مشاهده می‌شود مولفه سوم، به اندیس های آخر ضریب بیشتری اختصاص می‌دهد. از طرفی می‌دانیم نوکلئوبیس‌ها، واحدهای سازنده کروموزم‌ها هستند.همچنین کروموزم‌های آخر در مردان به صورت XY و در زنان به صورت XX می‌باشد؛ بنابراین مقادیر نوکلئوبیس‌های اندیس‌های آخر در مردان و زنان متفاوت است. درنتیجه مولفه سوم برای زنان و مردان مقدار بسیار متفاوتی را برمی‌گرداند. این مورد توجیه می‌کند که چرا در قسمت d ، مولفه سوم، برای تفکیک جنسیت به خوبی عمل می‌کند.

۵)

a) کد مربوط به این قسمت در فایل a.py قراردارد.مرکز هر کلاستر در هر مرحله، داده‌ای از آن کلاستر است که میانگین فاصله آن از سایر داده‌های همان کلاستر، کمینه باشد. خروجی با نام output.txt ذخیره‌شده‌است. ساختار فایل خروجی به این صورت است:

Centroid ID: [List of IDs]

****

...

b) کد مربوط به این قسمت در فایل b.py قرار دارد. برای انتخاب مراکز اولیه به این صورت عمل می‌کنیم:

۱) داده‌ای که بیشترین میانگین فاصله را از سایر داده‌ها دارد به عنوان اولین مرکز انتخاب و در آرایه output ذخیره می‌کنیم.

۲) داده‌ای که بیشترین میانگین فاصله را از داده‌های موجود در آرایه output دارد انتخاب می‌کنیم و به آرایه output اضافه می‌کنیم.

۳) مرحله ۲ تا زمانی که ۲۵ مرکز انتخاب شوند تکرار می‌کنیم.

خروجی که شامل ۲۵ مرکز اولیه است در فایل part_b.txt ذخیره شده است.

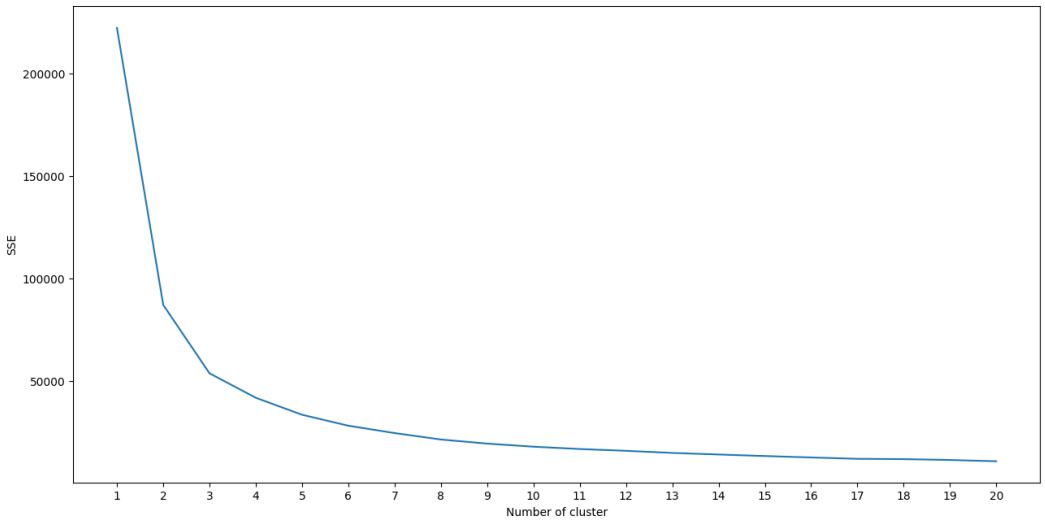
۶)

a) کد مربوط به این قسمت در فایل a.py قراردارد. ستون‌های مربوط به ’ID‘ و ’SubjectID‘ به علت بی‌ارتباط بودن حذف شده‌اند. همچنین برای جلوگیری از تاثیر بزرگی مقادیر ویژگی‌ها در مقدار کوواریانس، داده‌ها با استفاده از میانگین و انحراف معیار نرمال شده‌اند. در انتها قدرمطلق مقادیر کوواریانس محاسبه شده و ۱۱ مقدار اول آن چاپ شده‌اند.(ردیف اول مربوط به خود ’ALSFRS_slope‘ می‌باشد)

```
ALSFRS_slope          1.000000
ALSFRS_Total_range    0.819305
trunk_range           0.686407
hands_range           0.632350
ALSFRS_Total_min      0.601810
leg_range             0.584507
mouth_range           0.553194
trunk_min             0.488530
mouth_min             0.435538
respiratory_range     0.432878
hands_min             0.424107
```

بنابراین ویژگی‌های فوق(به جز ردیف اول) به عنوان ویژگی‌های دیتاست کاهش‌یافته انتخاب می‌شوند.

b) کد مربوط به این قسمت در فایل b.py قراردارد. بدین منظور، نمودار مقدار SSE برحسب K را رسم می‌کنیم:

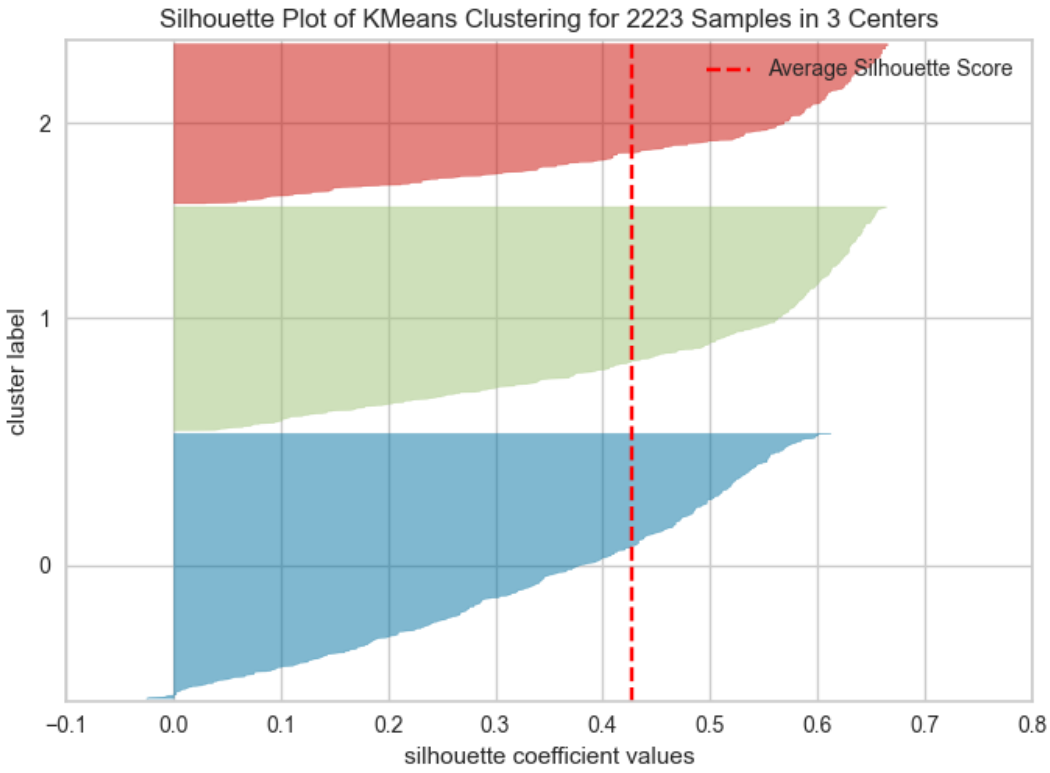


با توجه به نمودار فوق و روش Elbow، به نظر می‌رسد 3 مقدار مناسبی برای پارامتر K می‌باشد.

c) کد مربوط به این قسمت در فایل c.py قراردارد. ابتدا مرکز کلاسترها رو نمایش می‌دهیم:

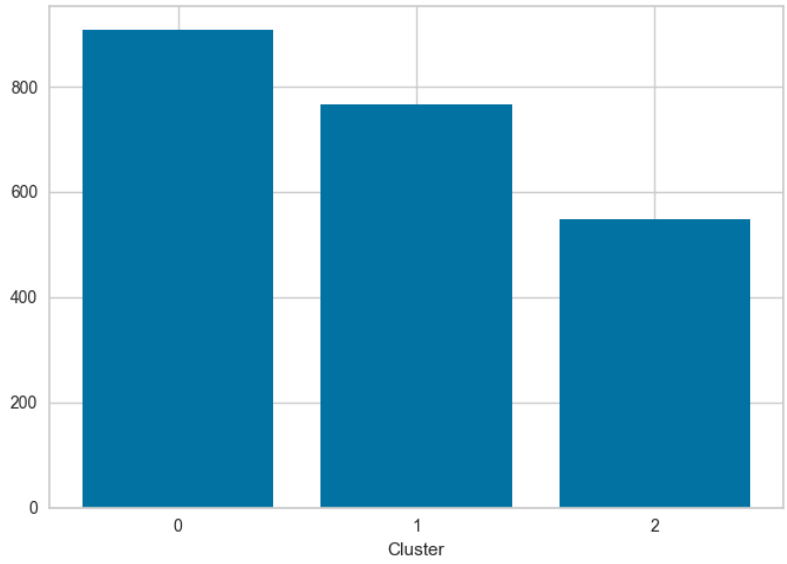
	ALSFRS_Total_range	trunk_range	hands_range	ALSFRS_Total_min	leg_range	mouth_range	trunk_min	mouth_min	respiratory_range	hands_min
0	0.027224	0.007904	0.007931	18.611233	0.006500	0.005856	2.313877	8.227974	0.002570	2.325991
1	0.012945	0.004110	0.003669	29.367666	0.004202	0.002713	5.352021	10.148631	0.001282	5.537158
2	0.042387	0.010098	0.009643	8.691606	0.008348	0.013250	0.666058	3.715328	0.004143	0.757299

برای رسم نمودار silhouette از کتابخانه YellowBricks استفاده می‌کنیم:

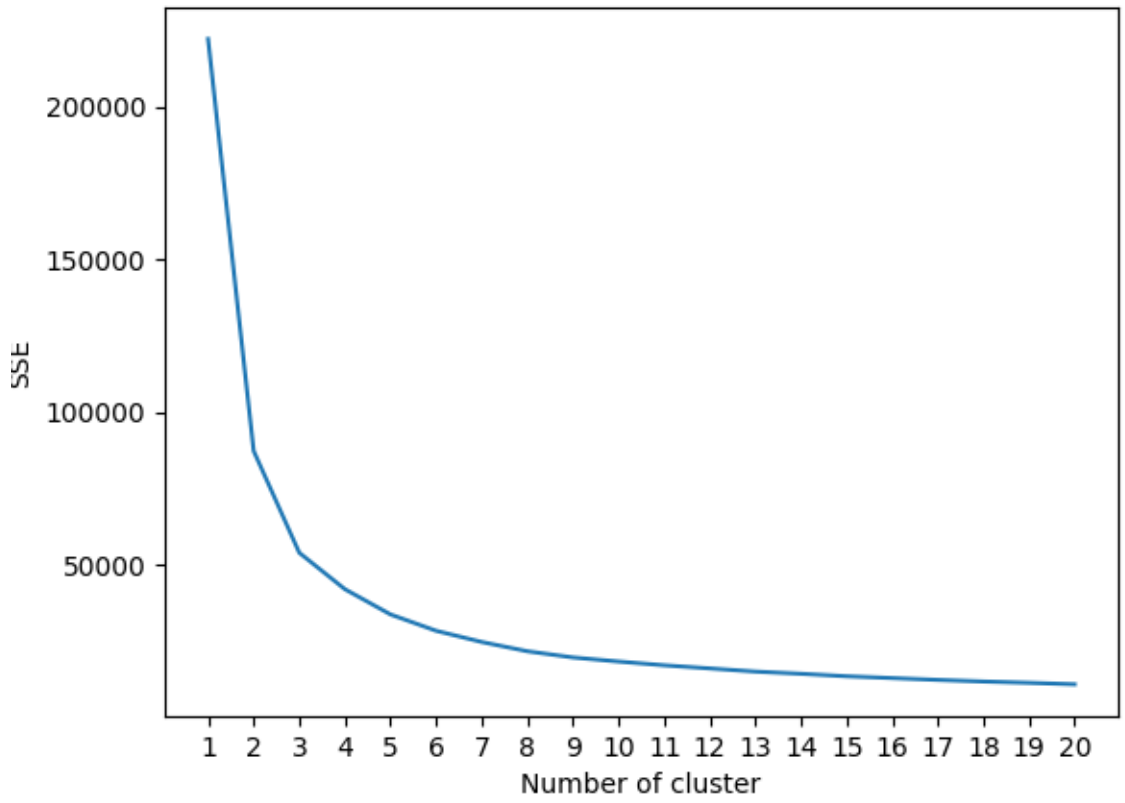


ضریب silhouette نشان دهنده فاصله هر نمونه از کلاستر همسایه است. مقدار این ضریب می‌تواند از 1 تا 1- متغیر باشد. مقدار 1 نشان می‌دهد که نمونه از کلاستر همسایه بسیار دور است؛(و بنابراین احتمالا به درستی دسته بندی شده) مقدار صفر نشان می‌دهد که نمونه روی مرز بین دو کلاستر همسایه قراردارد و درنهایت مقادیر منفی یعنی ممکن است آن نمونه به کلاستر اشتباه منتسب شده‌باشد. براساس این توضیحات دسته‌بندی فوق مناسب به نظر می‌رسد؛ زیرا اکثر داده ها در سمت راست خط میانگین(خط چین قرمز) قرار دارند و بنابراین از کلاسترهای همسایه فاصله مناسبی دارند.

در انتها تعداد اعضای هر کلاستر را در نمودار زیر نشان می‌دهیم:



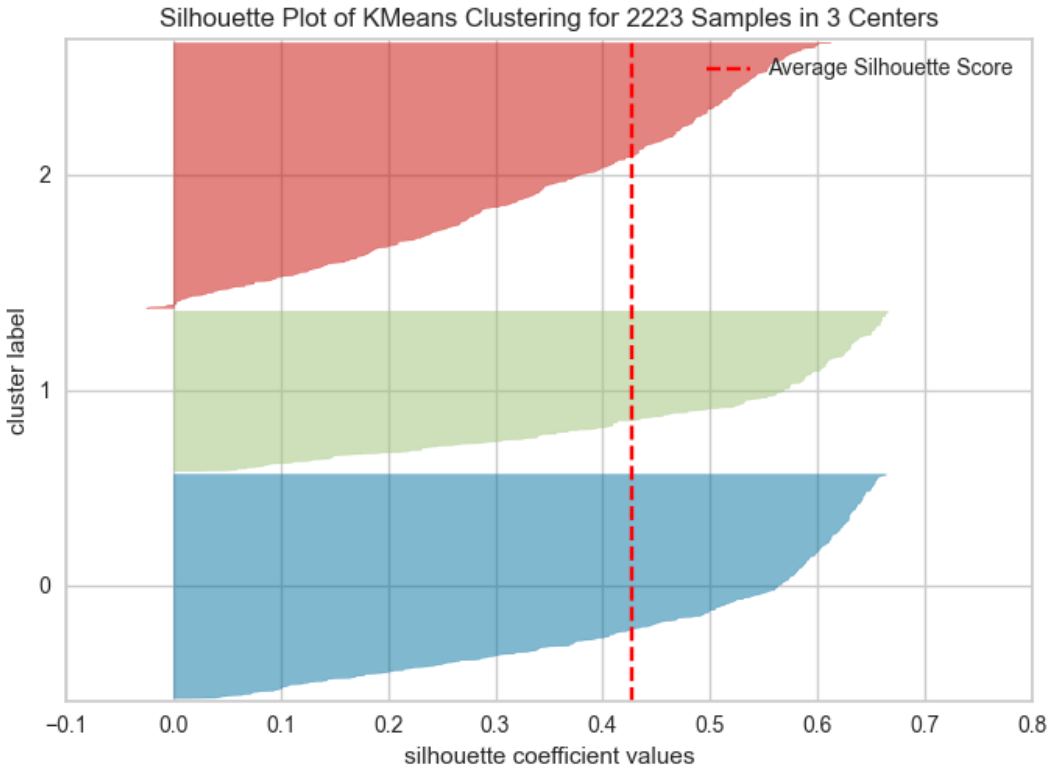
d) کد مربوط به این بخش در فایل d.py قراردارد. همانند بخش b عمل می‌کنیم، با این تفاوت که نقاط اولیه کلاسترها به وسیله روش k-mean++ تعیین می‌شوند.

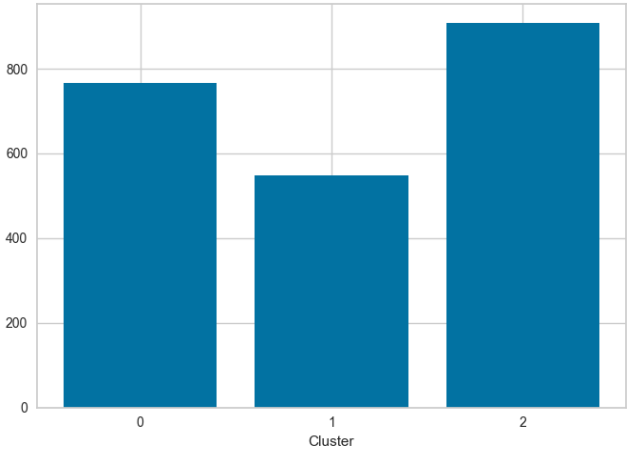


مجددا مقدار ۳ برای پارامتر k انتخاب می‌شود.

e) کد مربوط به این بخش در فایل e.py قراردارد. نتایج در این بخش تفاوتی با بخش c ندارد؛ تنها شماره کلاسترها متفاوت است. مجددا نتایج را نمایش می‌دهیم:

	ALSFRS_Total_range	trunk_range	hands_range	ALSFRS_Total_min	leg_range	mouth_range	trunk_min	mouth_min	respiratory_range	hands_min
0	0.012945	0.004110	0.003669	29.367666	0.004202	0.002713	5.352021	10.148631	0.001282	5.537158
1	0.042387	0.010098	0.009643	8.691606	0.008348	0.013250	0.666058	3.715328	0.004143	0.757299
2	0.027224	0.007904	0.007931	18.611233	0.006500	0.005856	2.313877	8.227974	0.002570	2.325991





باتوجه به تحلیلی که در بخش c ارائه شد، نتایج الگوریتم رضایت بخش است.

f) کد مربوط به این بخش در فایل f.py قراردارد. مجددا برای رسم نمودار silhouette از کتابخانه YellowBricks استفاده می‌کنیم. در صورت اجرای این کتابخانه برای مدل‌های‌سلسه‌مراتبی، با خطای زیر مواجه می‌شویم:

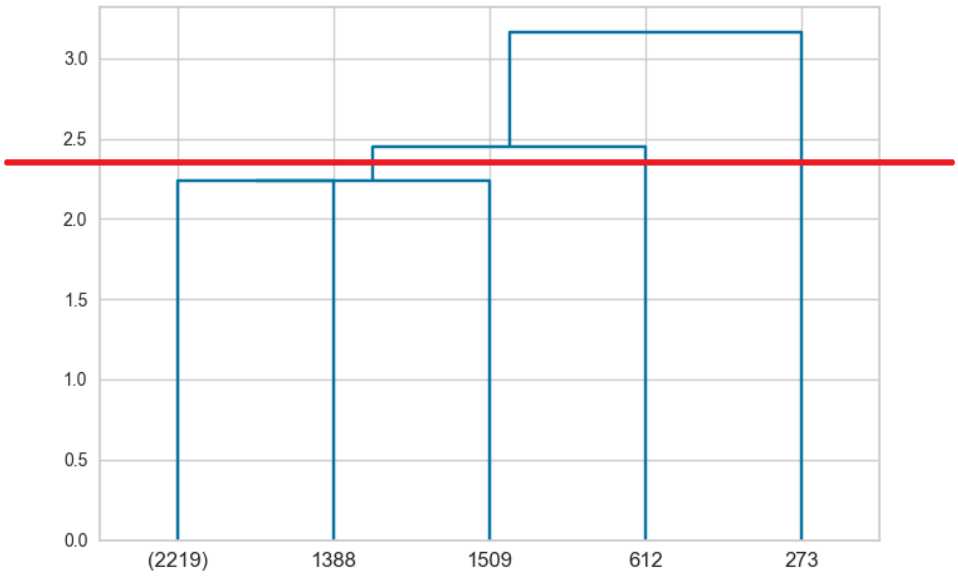
```
  File "C:\Users\acer\AppData\Local\Programs\Python\Python39\lib\site-packages\yellowbrick\cluster\silhouette.py", line 145, in fit
    labels = self.estimator.predict(X)
AttributeError: 'AgglomerativeClustering' object has no attribute 'predict'
```

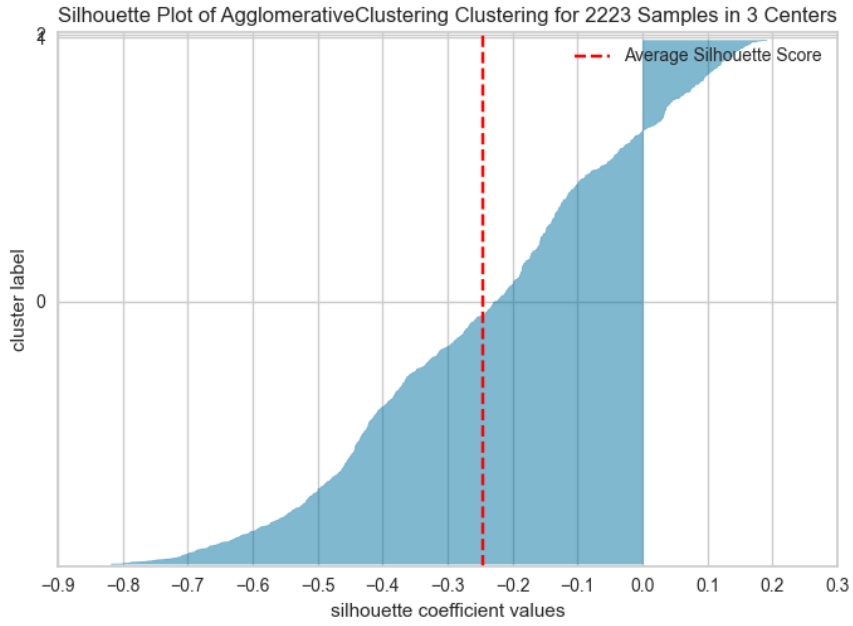
دلیل خطای فوق این است که مدل AgglomerativeClustering در کتابخانه sklearn فاقد متد predict می‌باشد. برای حل این مشکل خط ۱۴۵ از فایل مورد اشاره در خطای بالا را به این صورت تغییر می‌دهیم:

```
144         # Compute the scores of the cluster
145         try:
146             labels = self.estimator.predict(X)
147         except:
148             labels = self.estimator.fit_predict(X)
```

همچنین برای رسم نمودار دندوگرام از کتابخانه scipy و تابع ارائه شده در مستندات آن استفاده می‌کنیم. جهت مقایسه بهتر با نتایج قسمت های قبل، تعداد کلاسترها را ۳ در نظر می‌گیریم:

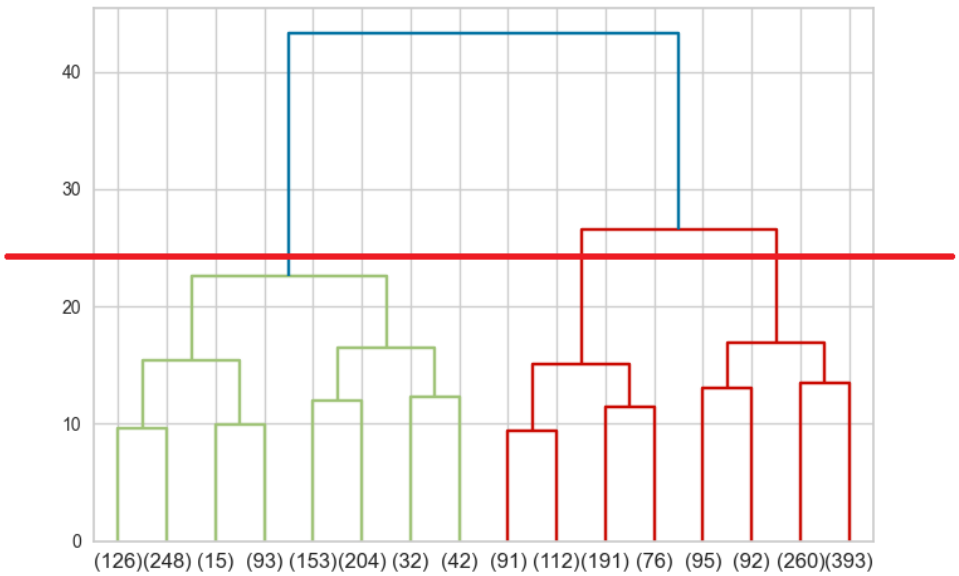
ابتدا به بررسی single linkage می‌پردازیم:

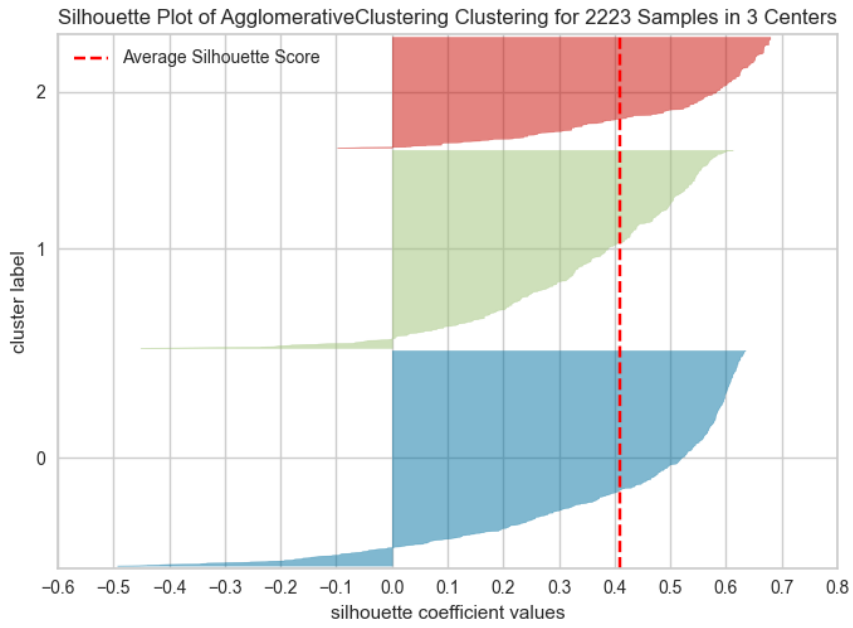




در نمودار دندوگرام، اعداد داخل پرانتز نشان‌دهنده تعداد داده‌های موجود در آن نقطه، و اعداد بدون پرانتز نشان‌دهنده اندیس آن داده است. با توجه به نمودارهای فوق، این الگوریتم عملکرد مناسبی نداشته است. همچنین در نمودار silhouette تنها منحنی مربوط به کلاستر ۰ قابل مشاهده است؛ زیرا دو کلاستر دیگر تنها دارای یک عضو هستند. لازم به ذکر است افزایش تعداد کلاسترها نیز تاثیری در بهبود عملکرد این الگوریتم ندارد.

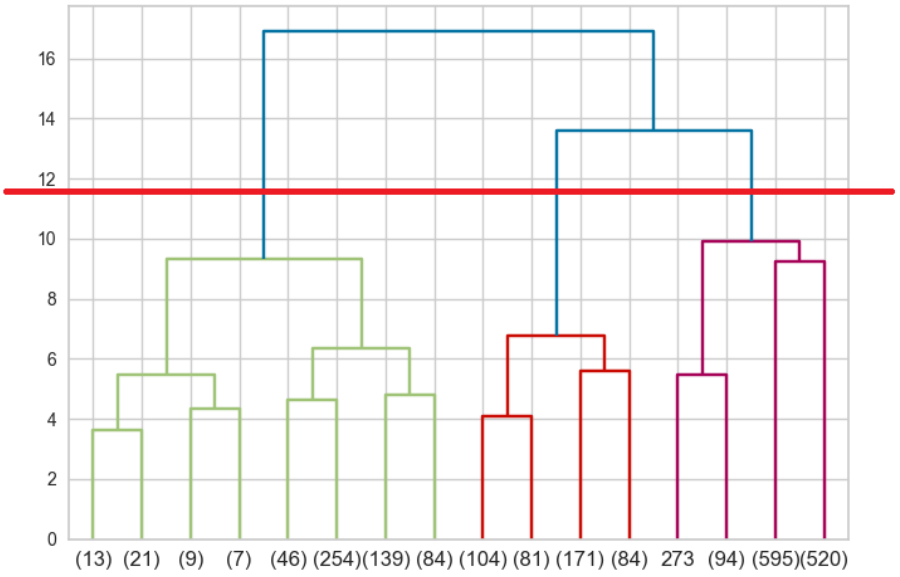
در ادامه الگوریتم complete-linkage را بررسی می‌کنیم:

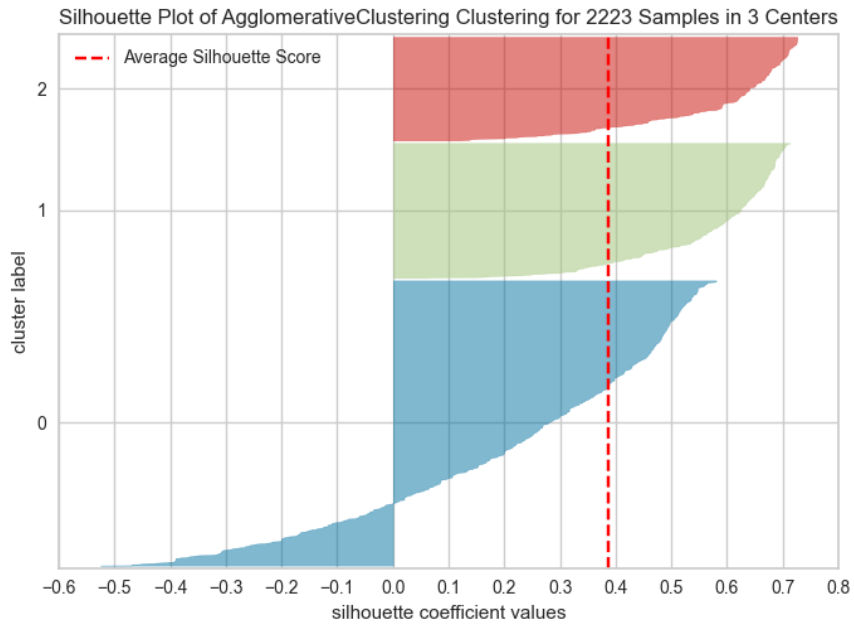




همانطور که مشاهده می‌شود، این الگوریتم عملکرد بسیار بهتری نسبت به الگوریتم single دارد. با توجه به نمودار silhouette، تنها مقدار کمی از داده ها دارای ضریب منفی هستند و اکثر آن‌ها ضریبی بیش از مقدار میانگین دارند.

در انتها نتایج روش average-linkage، را بررسی می‌کنیم:





با توجه به نمودار بالا، این روش کلاستر ۰ را به خوبی ایجاد نکرده است ولی نتایج آن همچنان نسبت به الگوریتم single-linkage بسیار بهتر است.

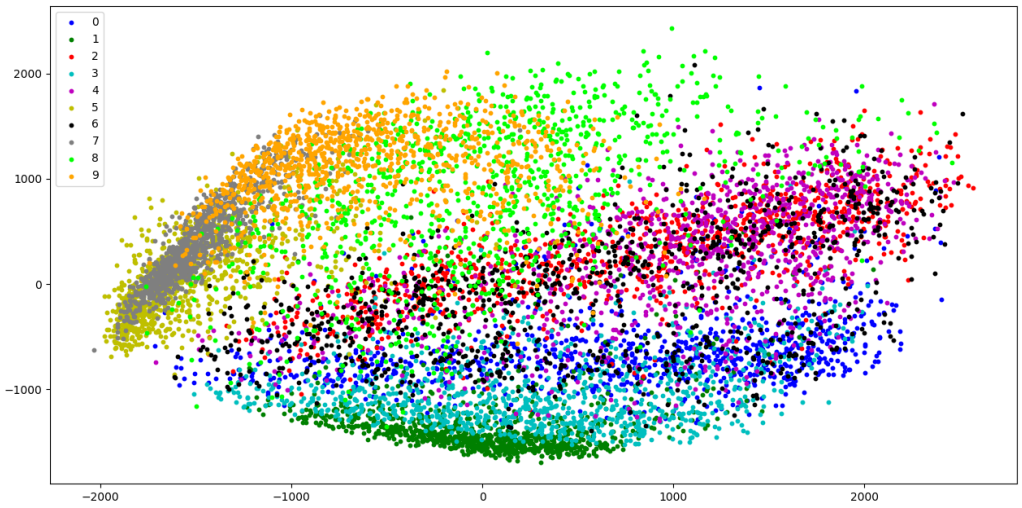
g) با بررسی نتایج بخش‌های قبل و بویژه نمودارهای silhouette، الگوریتم k-mean بهترین نتیجه را برای دسته‌بندی داده‌ها داشته است.

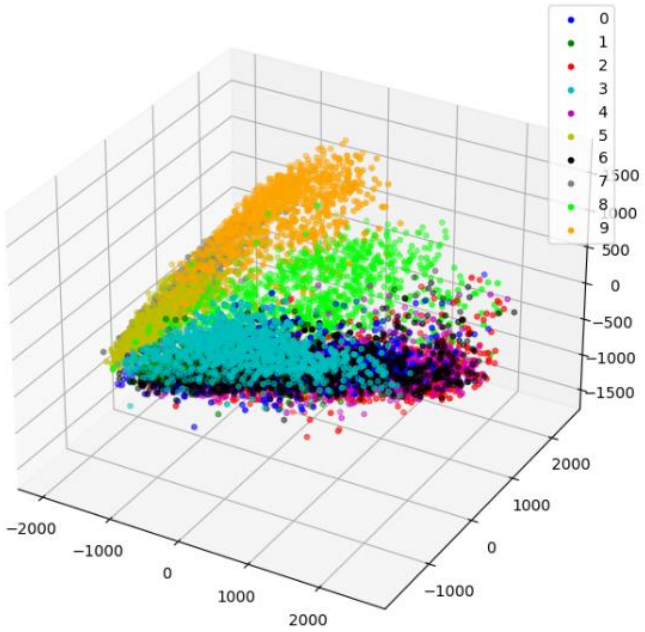
۷)

a) کد مربوط به این بخش در فایل a.py قراردارد. ابتدا به ۲۰ مقدارویژه اول ماتریس داده‌ها را حساب می‌کنیم:

```
[1288319.52477778  779197.62253773  265730.43854769  218669.76933454
  169257.23458071  152452.76424582  104674.41864859   83982.28146169
   58343.40693482   57195.68412973   43687.97422622   40723.98957408
   33555.72295231   28600.91358106   27417.80883762   25851.07030097
   24625.23833536   23565.2096588    21246.05528271   19616.18064924]
```

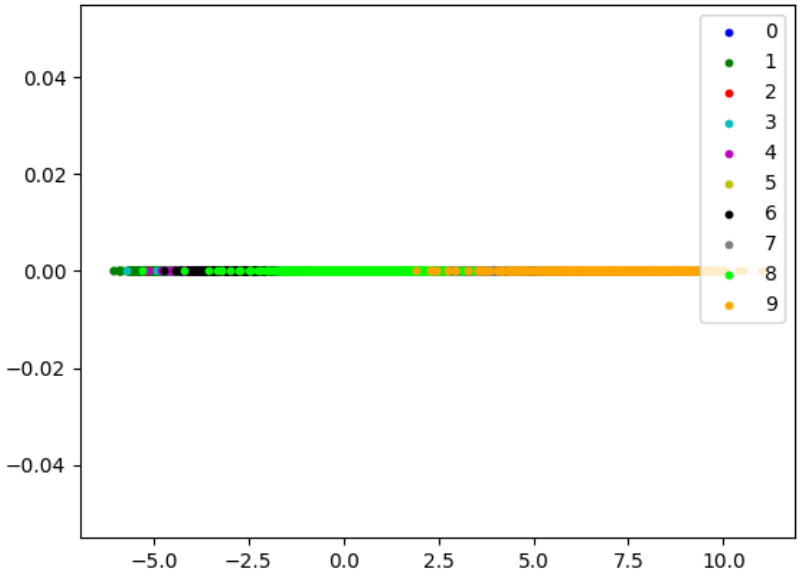
با توجه به مقادیر فوق، واریانس داده‌ها در جهت‌های مختلف نسبتاً زیاد است؛ بنابراین ممکن است دو و سه مولفه اول برای دسته بندی‌داده ها کافی نباشد. به هرحال تصویر داده‌ها را روی دو و سه مولفه اول نمایش می‌دهیم:

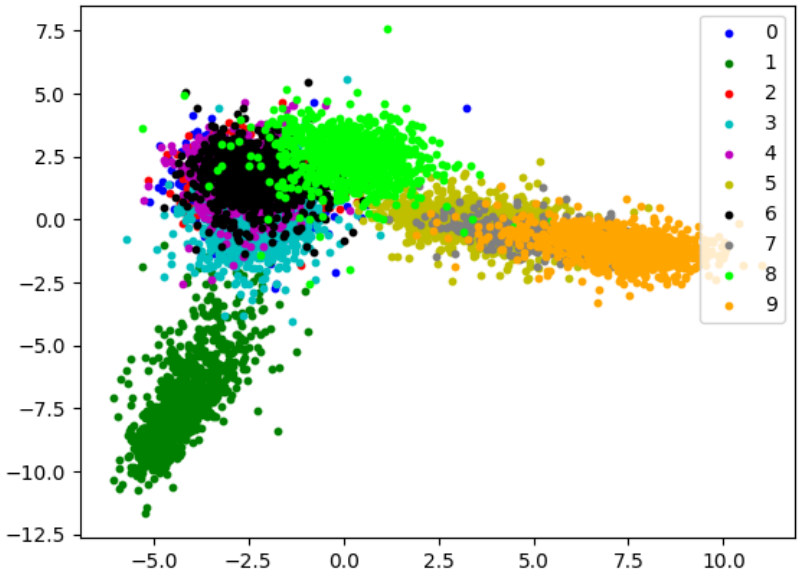




با توجه به شکل‌های فوق، دسته‌ها به خوبی از هم جدا نشده‌اند و حتی کلاس‌هایی که بهم شبیه هستند تقریبا برهم منطبق شده‌اند. اگرچه این نتایج با توجه به اینکه روش PCA یک روش کاهش بعد بدون نظارت است کاملا مورد انتظار بوده است.

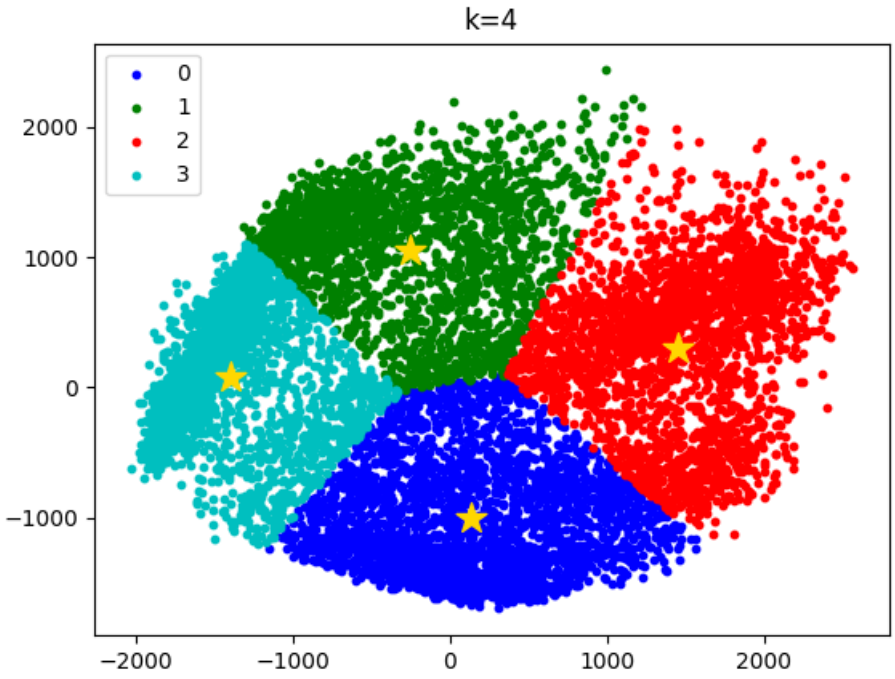
b) کد مربوط به این قسمت در فایل b.py قراردارد. داده‌ها را با روش LDA به فضای یک و دوبعدی منتقل می‌کنیم:

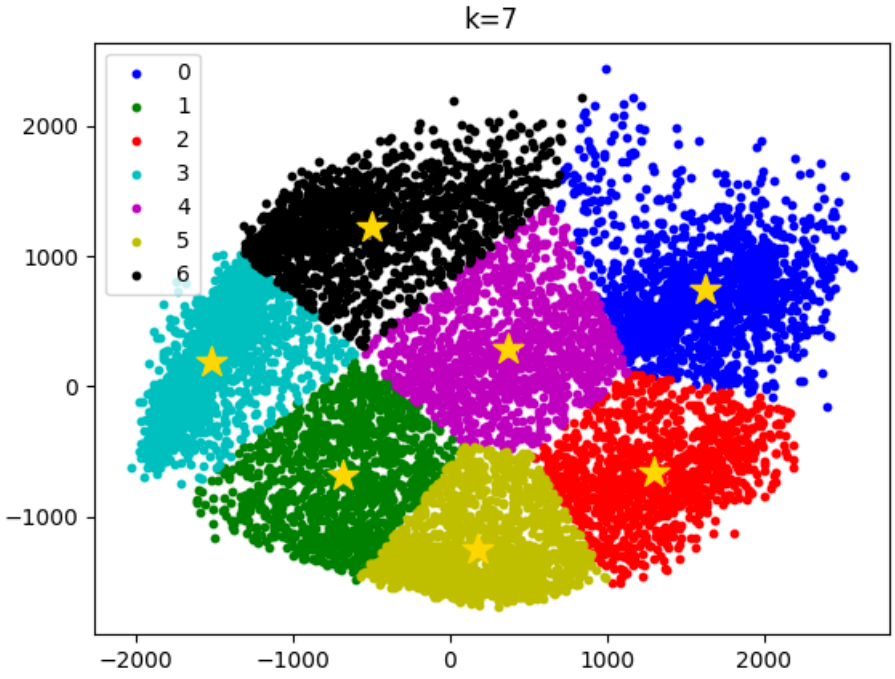


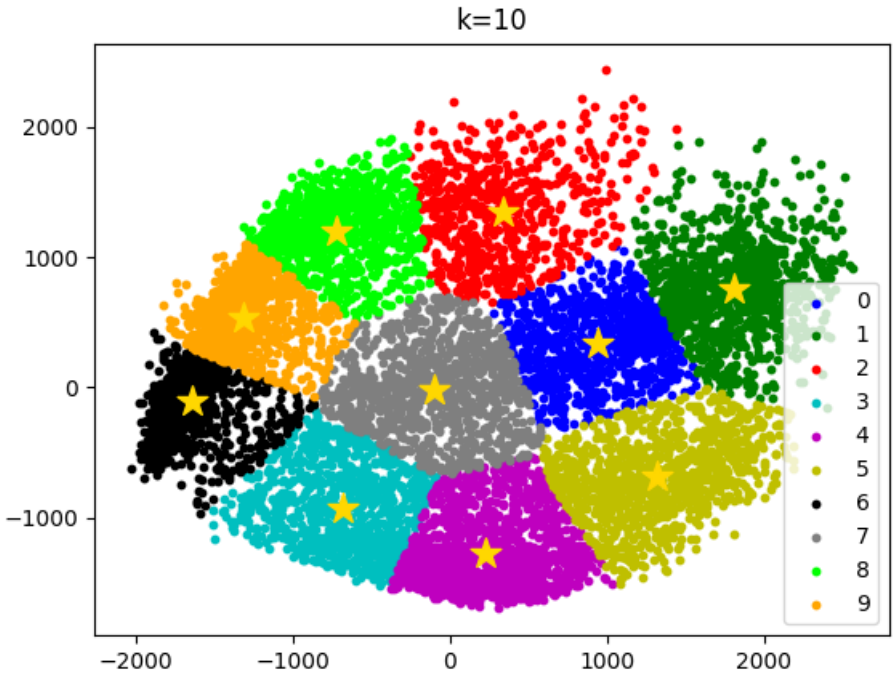


اگرچه نتایج همچنان به اندازه کافی رضایت‌بخش نیست، اما LDA به ویژه در حالت دوبعدی نسبت به PCA عملکرد نسبتاً بهتری دارد؛ زیرا LDA یک روش بانظارت است و برای کاهش بعد به برچسب داده‌ها توجه می‌کند.

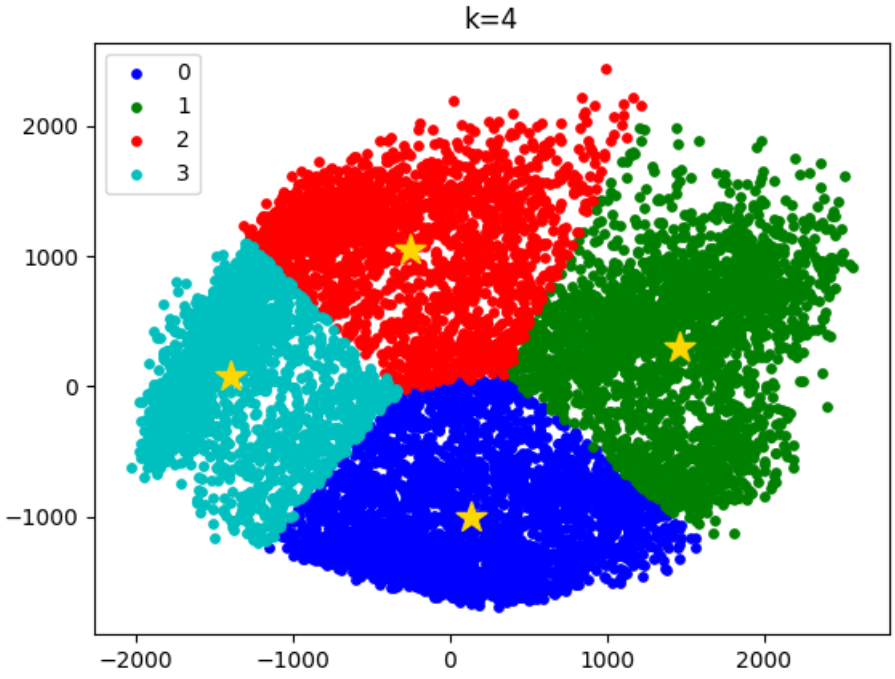
c) کد مربوط به این بخش در فایل c.py قراردارد. مطابق انتظار نتایج به هیچ‌وجه قابل قبول نیست و هیچ شباهتی به نتایج بخش a ندارد.

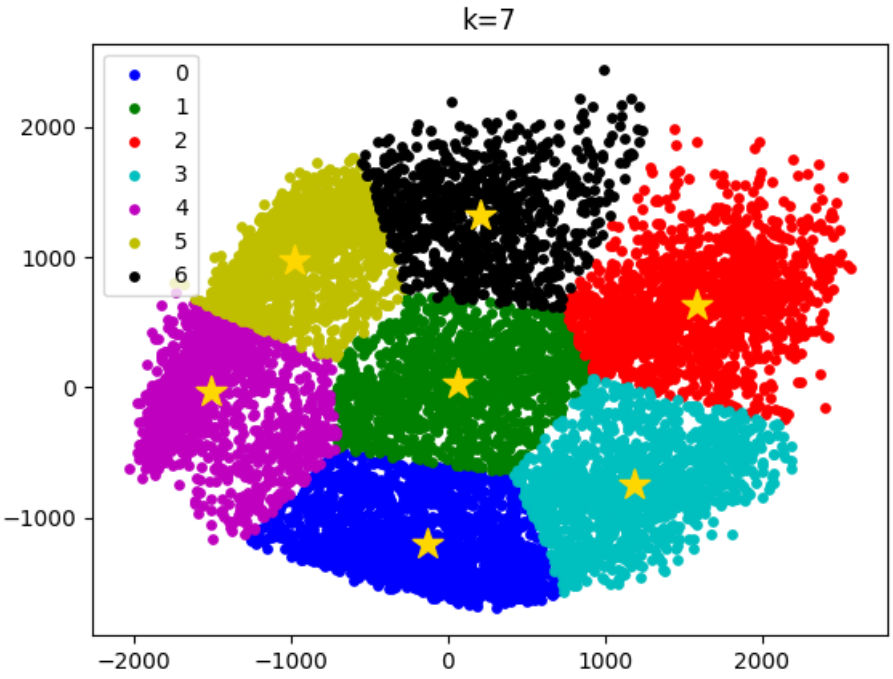

k=4
0
1
2
3
2000
1000
0
−1000
−2000
−1000
0
1000
2000

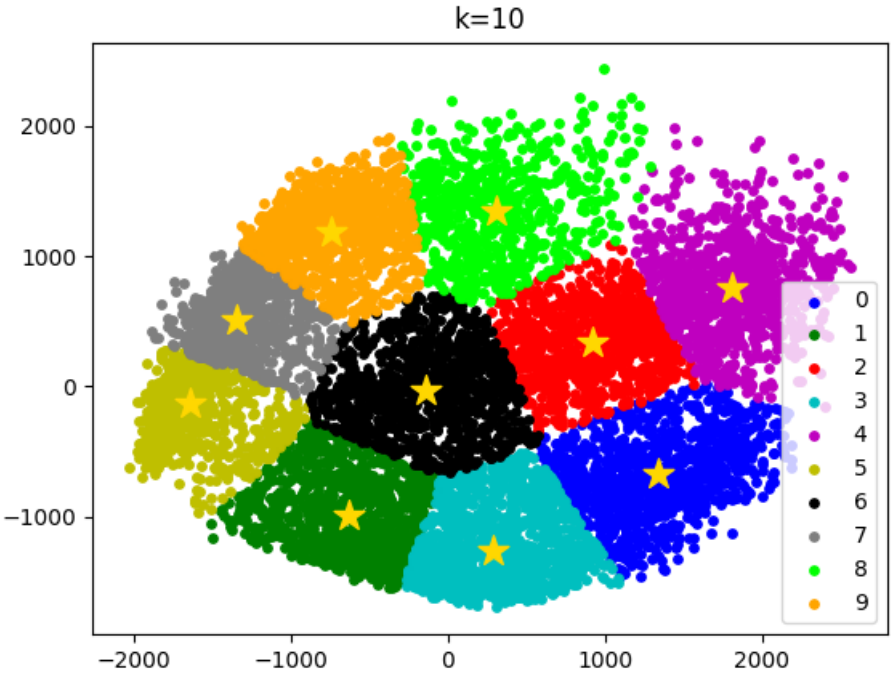

k=7
0
1
2
3
4
5
6
2000
1000
0
−1000
−2000
−1000
0
1000
2000



d) کد مربوط به این بخش درفایل d.py قراردارد. بعد از انتخاب مراکز اولیه طبق خواسته سوال، نتایج به این صورت است:

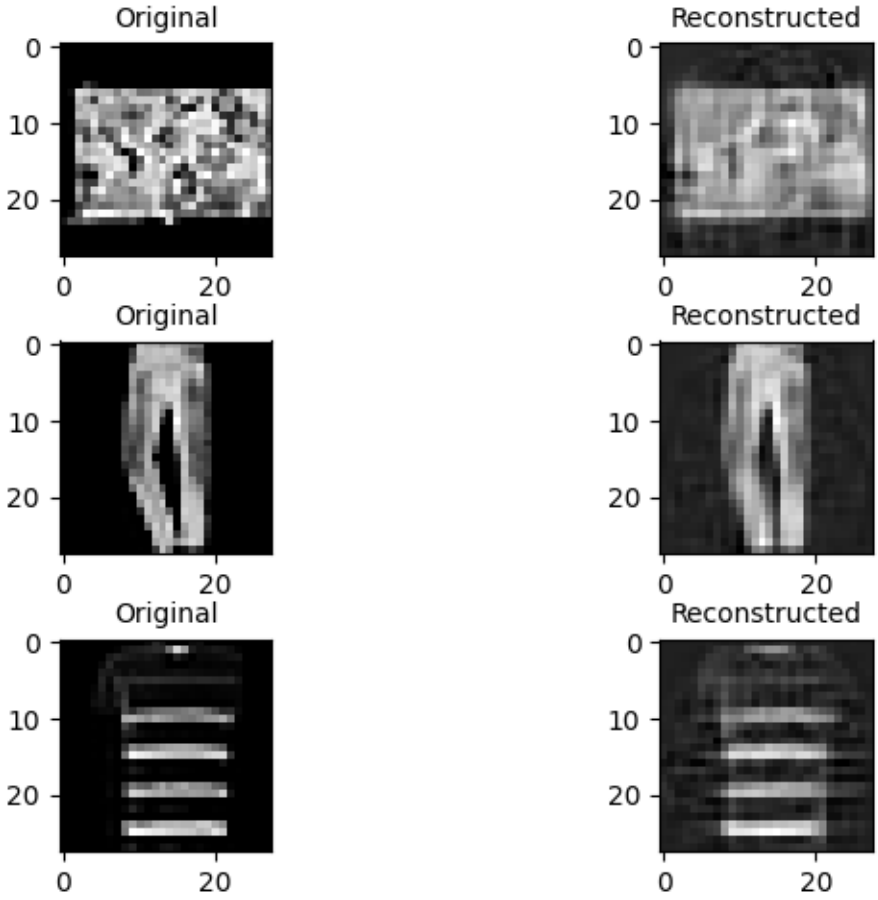






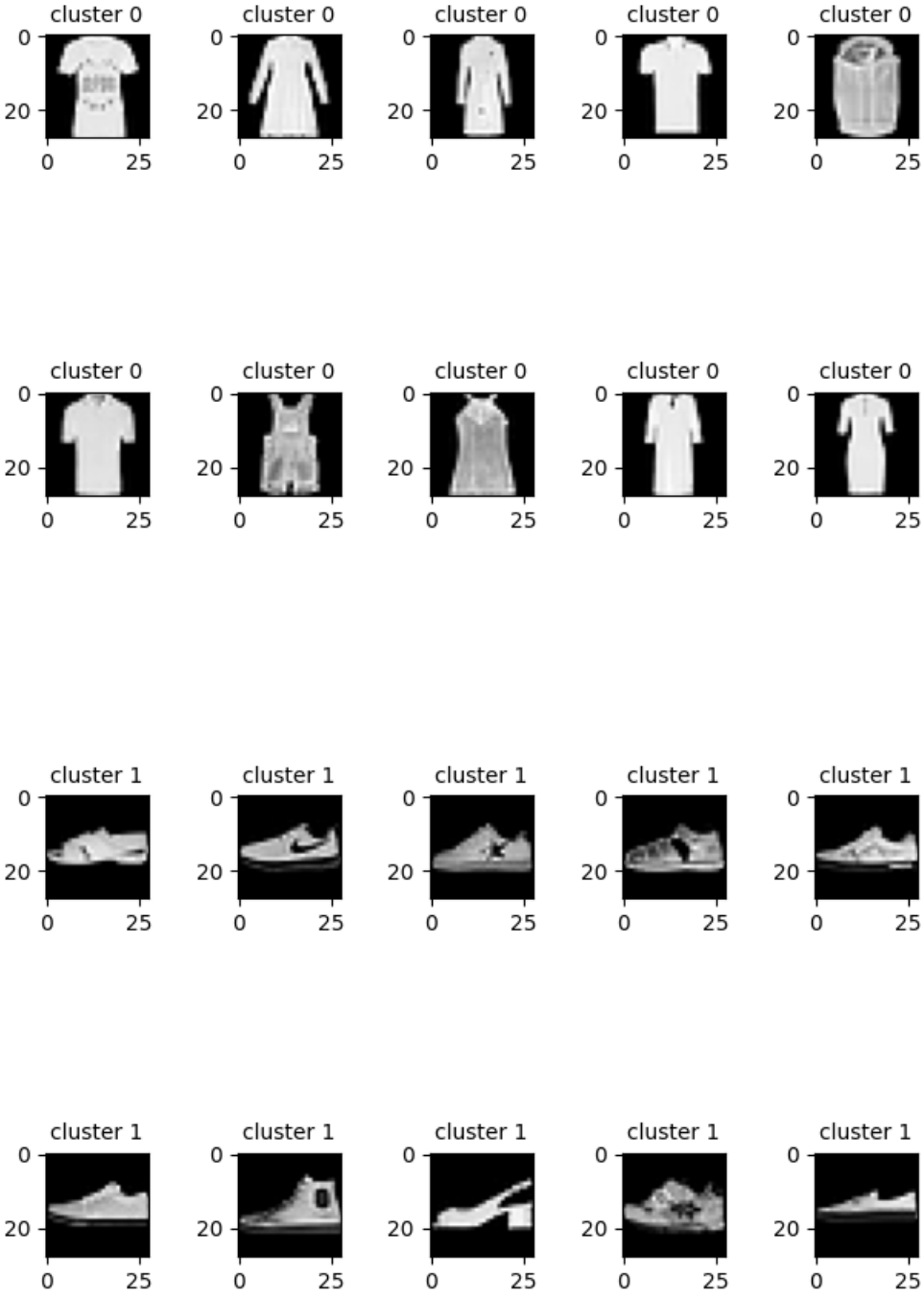
در مجموع به جز تغییری که در$k=7$ مشاهده می‌شود، نتایج مانند قسمت قبل است. به هرحال این نتایج همچنان قابل قبول نمی‌باشند.

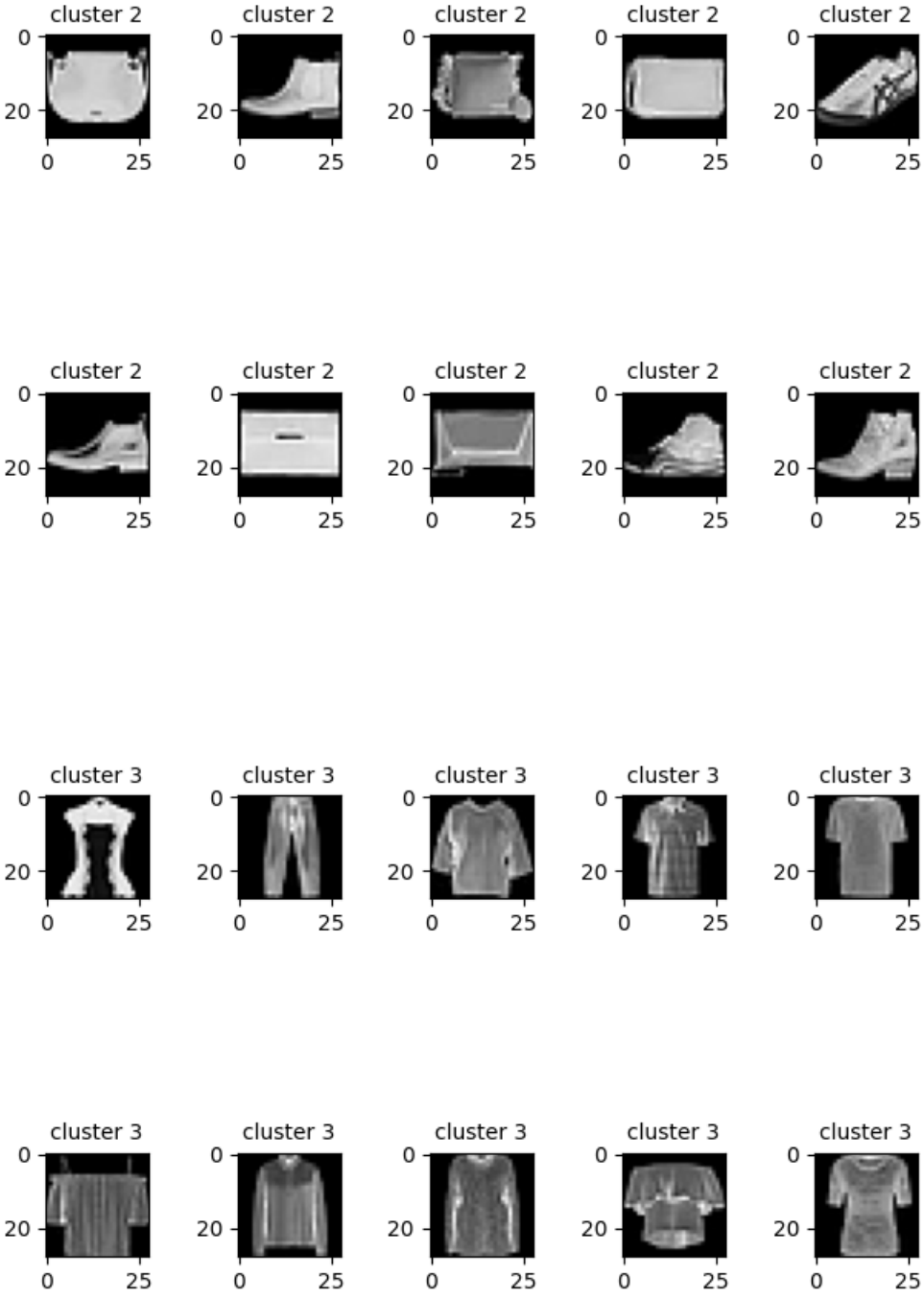
e) کد مربوط به این بخش در فایل e.py قرار دارد. تعداد ۱۸۵ مولفه اصلی، بدین منظور کافی می‌باشد. نتایج بازسازی سه تصویر تحت این مولفه‌ها در تصویر زیر ارائه شده است:

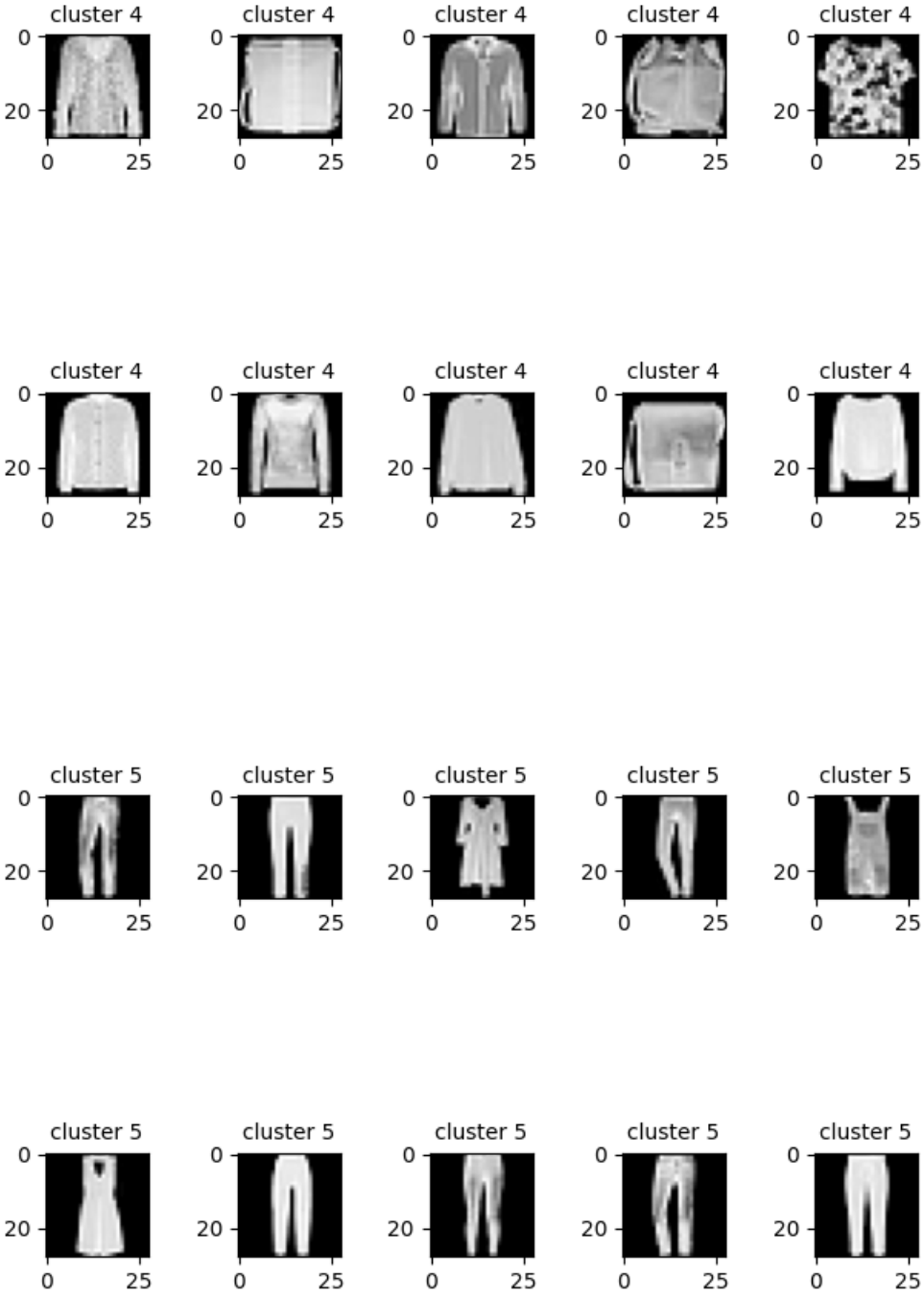


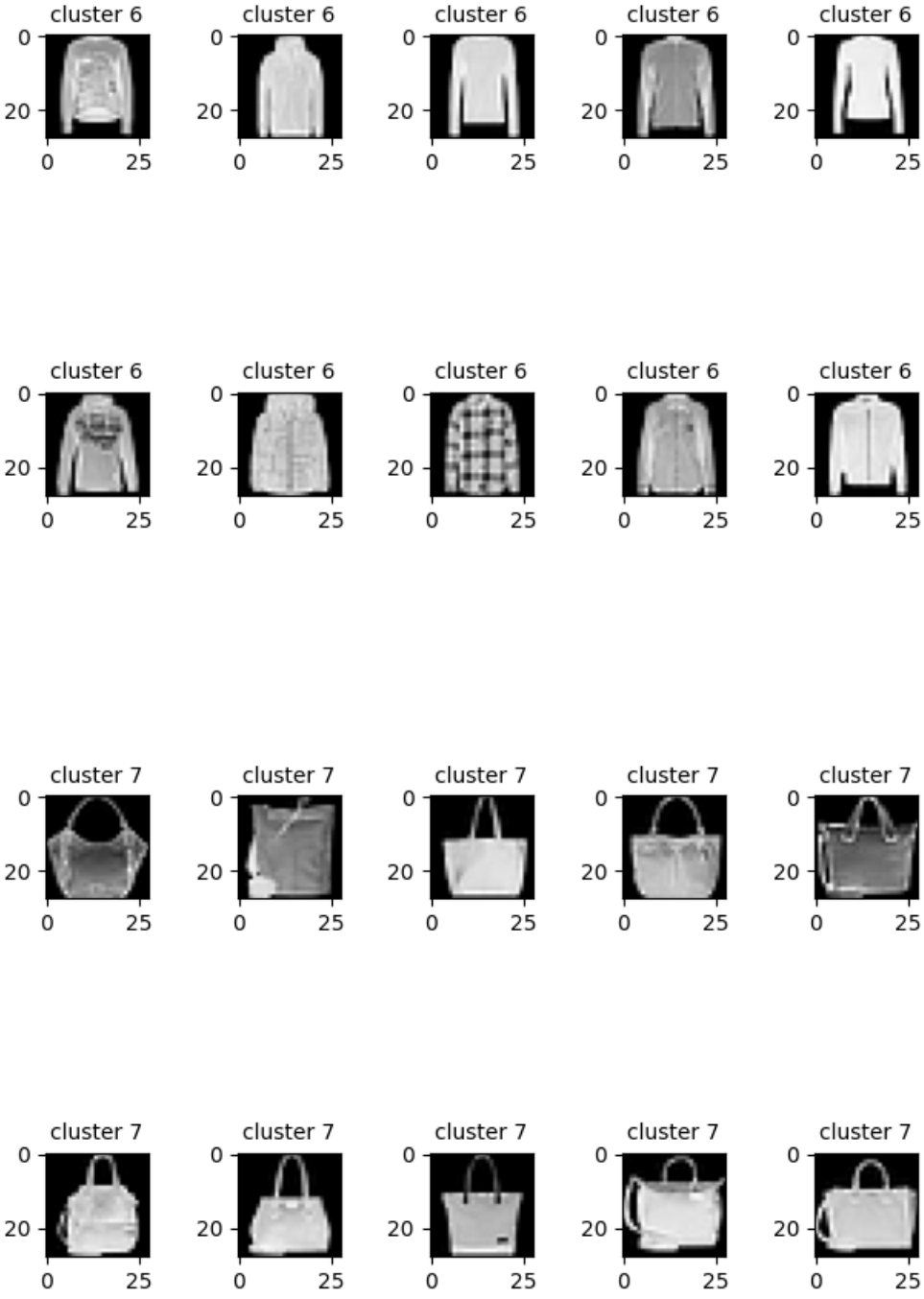
تحت این تبدیل، اگرچه برخی جزئیات از دست رفته‌است؛ اما شکل کلی اشیا و جزئیات اصلی آنها همچنان حفظ شده است.

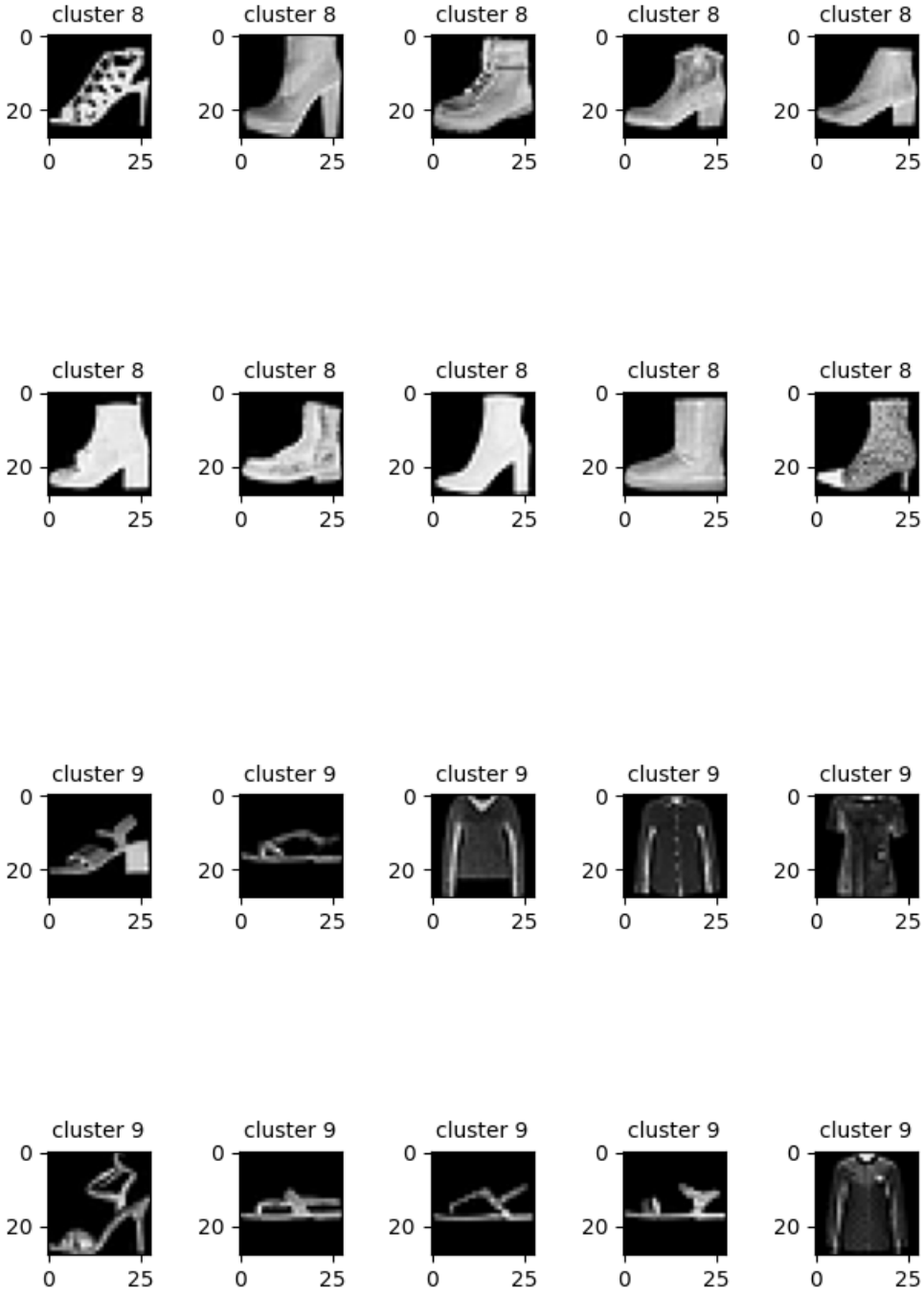
f) کد مربوط به این بخش در فایل f.py قراردارد. برای تنظیم نقاط اولیه از رویکرد k-means++ استفاده می‌کنیم. نتایج به این صورت است:


cluster 0
cluster 0
cluster 0
cluster 0
cluster 0
cluster 0
cluster 0
cluster 0
cluster 0
cluster 0
cluster 1
cluster 1
cluster 1
cluster 1
cluster 1
cluster 1
cluster 1
cluster 1
cluster 1
cluster 1
0
20
0
25


cluster 2
cluster 2
cluster 2
cluster 2
cluster 2
cluster 2
cluster 2
cluster 2
cluster 2
cluster 2
cluster 3
cluster 3
cluster 3
cluster 3
cluster 3
cluster 3
cluster 3
cluster 3
cluster 3
cluster 3
0
20
0
25

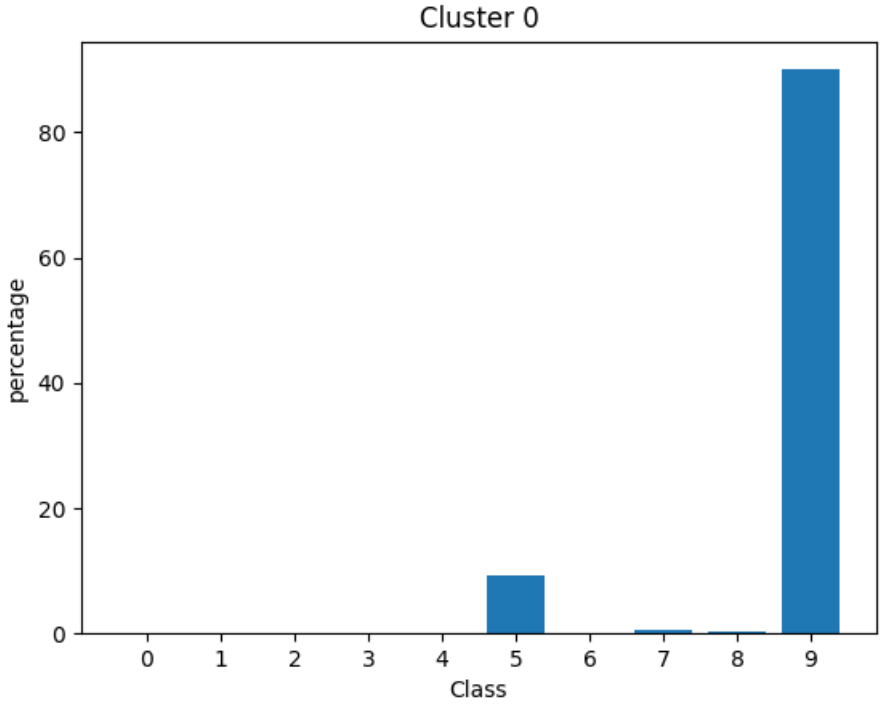

cluster 4
cluster 4
cluster 4
cluster 4
cluster 4
cluster 4
cluster 4
cluster 4
cluster 4
cluster 4
cluster 5
cluster 5
cluster 5
cluster 5
cluster 5
cluster 5
cluster 5
cluster 5
cluster 5
cluster 5
0
20
0
25

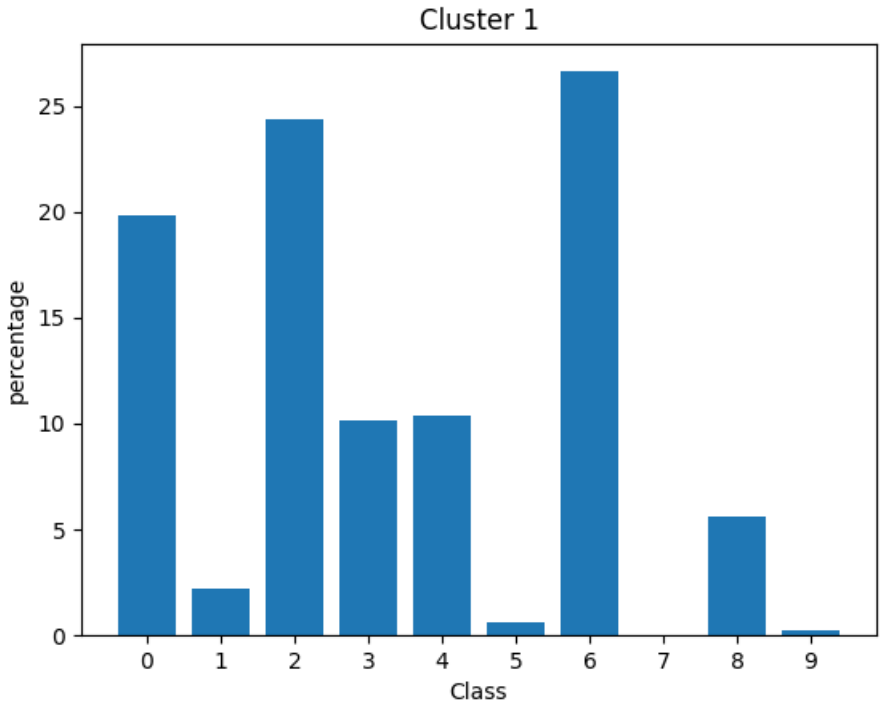

cluster 6
cluster 6
cluster 6
cluster 6
cluster 6
cluster 6
cluster 6
cluster 6
cluster 6
cluster 6
cluster 7
cluster 7
cluster 7
cluster 7
cluster 7
cluster 7
cluster 7
cluster 7
cluster 7
cluster 7
0
20
0
25

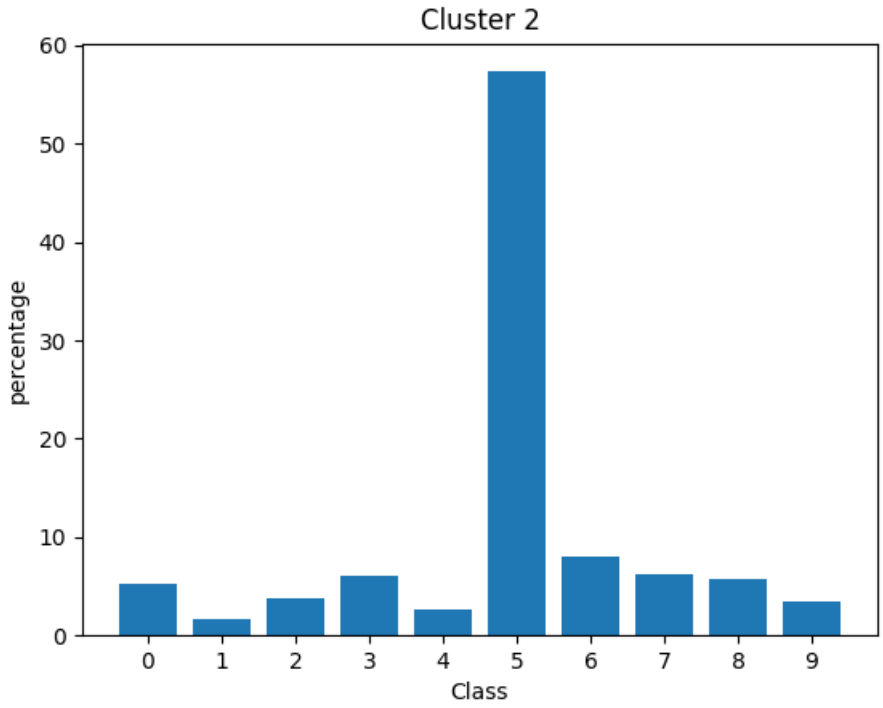


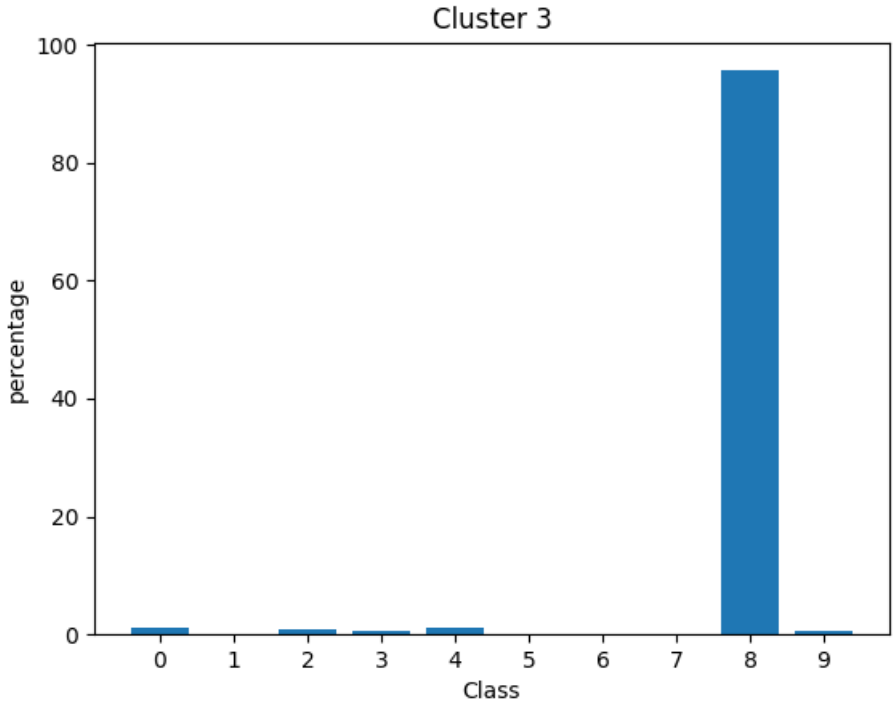
همانطور که مشاهده می‌شود اگرچه نتایج برای بعضی کلاسترها شامل مقدار قابل توجهی خطا می‌باشد؛ اما برای بعضی کلاسترها تقریبا رضایت بخش می‌باشد.

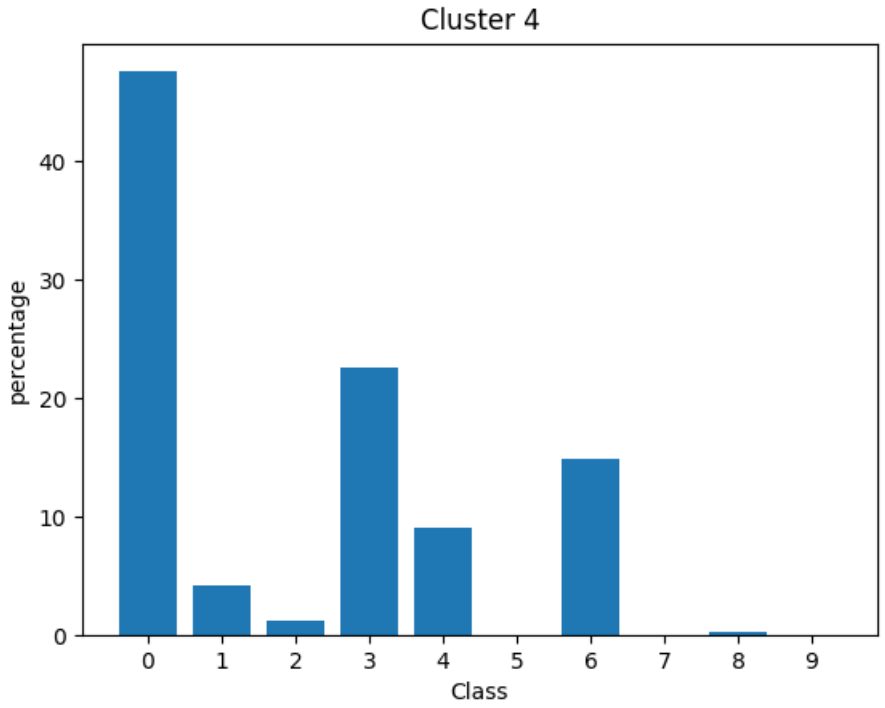
g) کد مربوط به این بخش در فایل g.py قراردارد. مجدداً از الگوریتم k-means++ استفاده می‌کنیم. لازم به ذکر است در اجراهای مختلف، ممکن است شماره کلاسترها تغییر کند؛ اما شکل کلی نمودارها ثابت و به این صورت است:

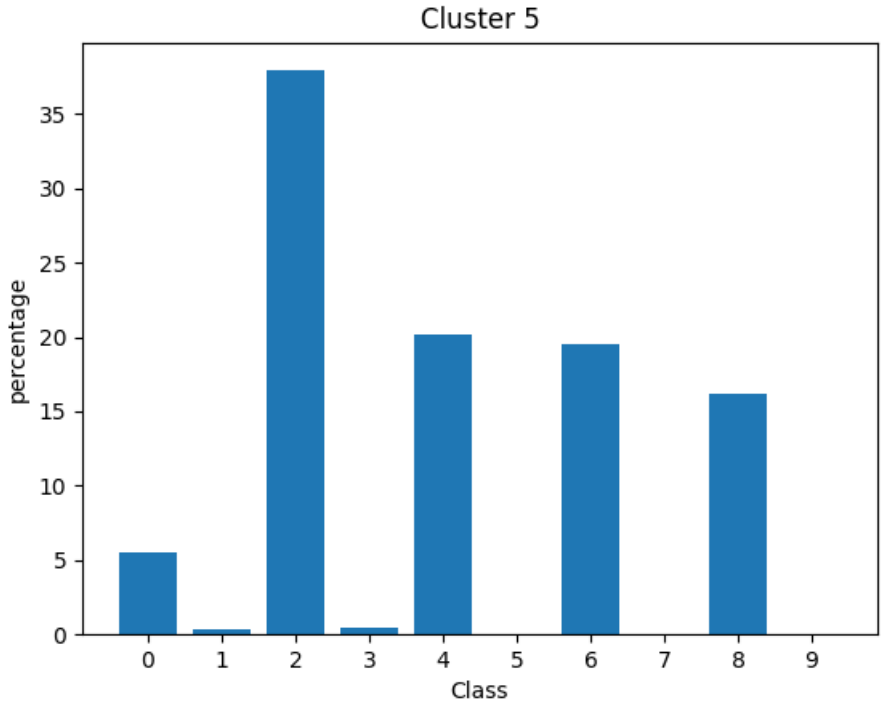


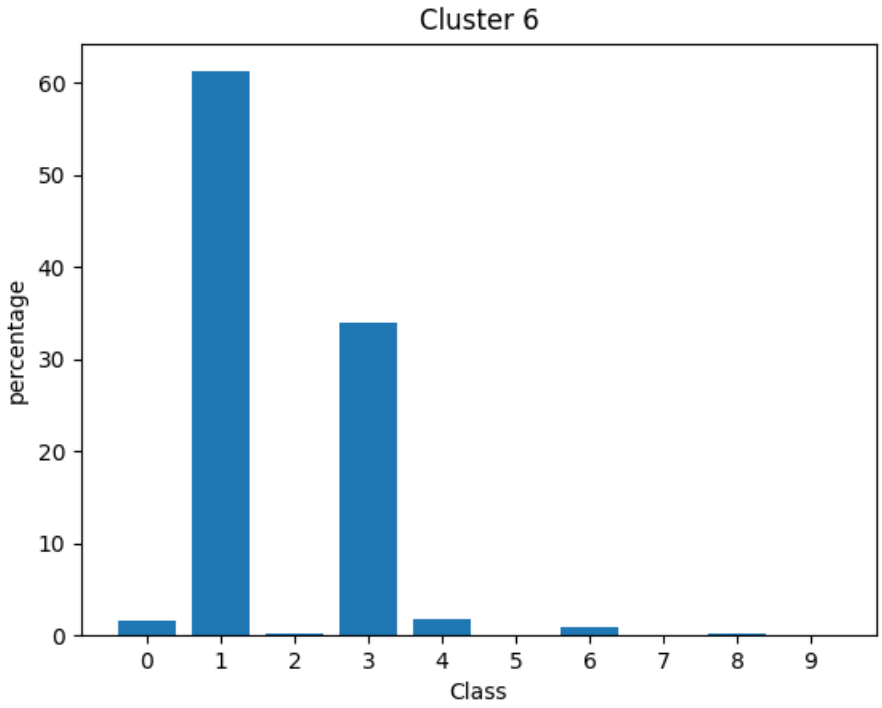


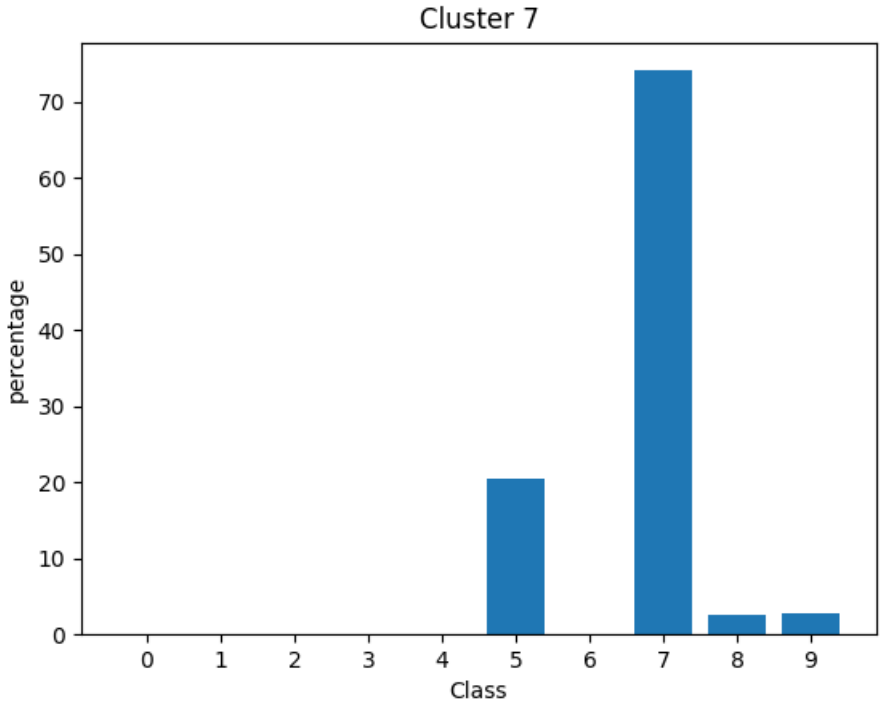

Cluster 2
percentage
Class
0
10
20
30
40
50
60
0
1
2
3
4
5
6
7
8
9

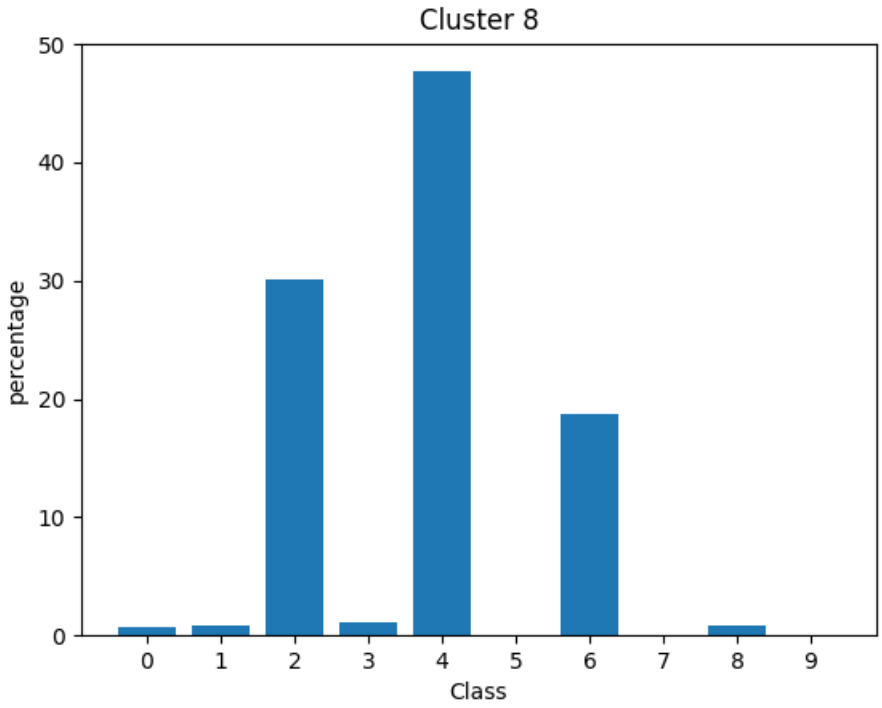

Cluster 3
percentage
Class
0
20
40
60
80
100
0
1
2
3
4
5
6
7
8
9

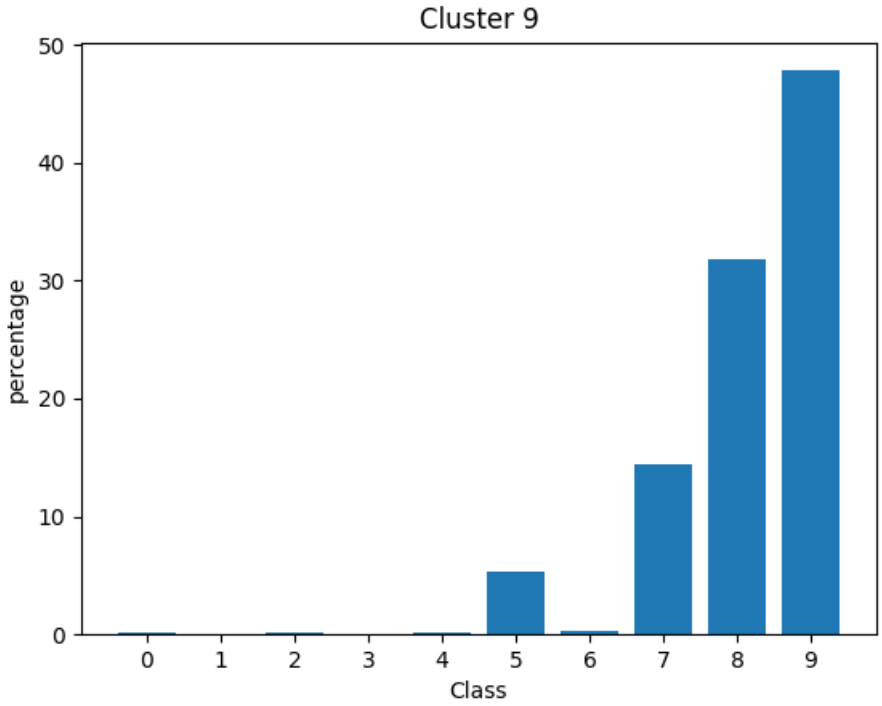

Cluster 4
percentage
0
10
20
30
40
0
1
2
3
4
5
6
7
8
9
Class


Cluster 5
percentage
0
5
10
15
20
25
30
35
0
1
2
3
4
5
6
7
8
9
Class


Cluster 6
percentage
0
10
20
30
40
50
60
0
1
2
3
4
5
6
7
8
9
Class


Cluster 7
percentage
0
10
20
30
40
50
60
70
0
1
2
3
4
5
6
7
8
9
Class



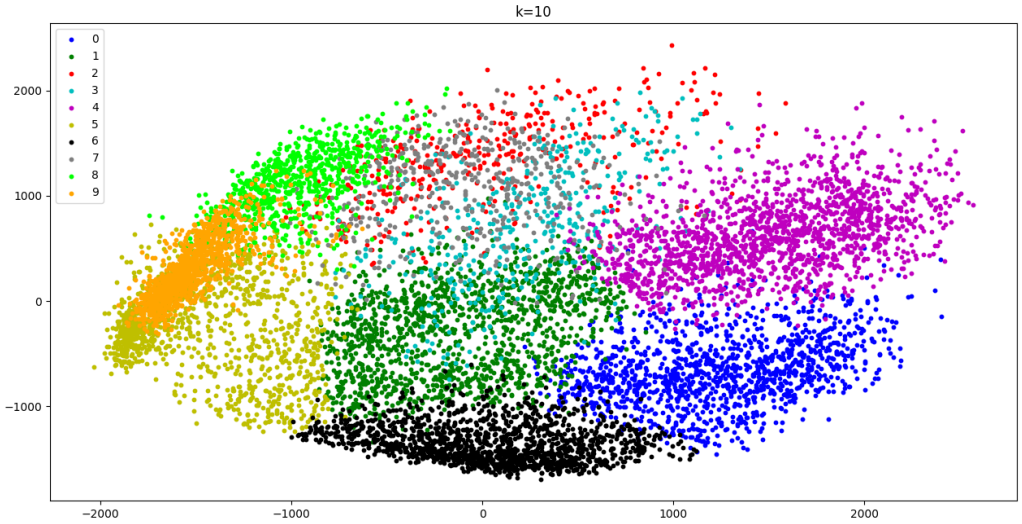


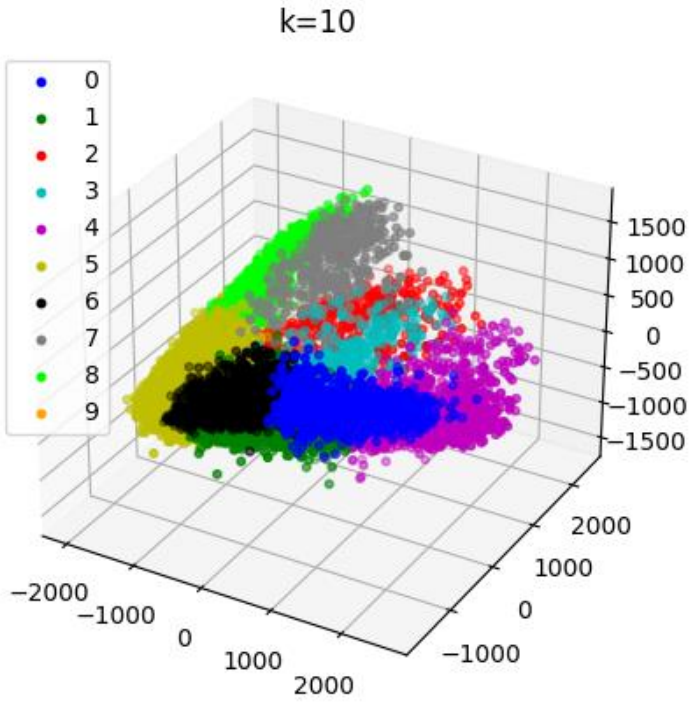
با توجه به تصاویر فوق:

۱) تقریبا تمامی کلاسترها شامل مقداری خطا می‌باشند؛ که البته برای برخی کمتر و برای بقیه بیشتر است.

۲) تقریبا هیچ کلاسی وجود ندارد که داده‌های مربوط به آن تماماً در یک کلاستر قرار گرفته باشد.

h) کد مربوط به این بخش در فایل h.py قرار دارد.





نتایج این بخش به نمودارهای بخش a نزدیک است؛ هرچند مقدار قابل توجهی از خطا نیز در آن مشهود است.

۸)

a) از دید عملکردی کاربرد PCA برای کاهش بعد است و نه افزایش آن. همچنین از دید جبرخطی، تعداد مولفه‌های‌اصلی که همان بردارهای ویژه هستند، حداکثر برابر با بعد داده‌ها است.

b) تصاویر ذاتاً دارای ابعاد بسیار بالا هستند. برای کاهش ابعاد تصاویر می توان از PCA به روش معمول استفاده کرد. برای بازسازی تصاویر در فضای اصلی(جهت نمایش و...) به این صورت عمل می‌کنیم:

$$X_{\text{reconstructed}} = \mu + X_T V^{-1}$$

در رابطه بالا μ بردار میانگین، X_T تصاویر کاهش یافته و V ماتریس تبدیل(شامل بردارهای ویژه) است.

c) بله؛ برای حل مشکل خطی بودن k-means می‌توان از شگرد kernel استفاده کرد. بدین منظور تابع kernel باید ویژگی‌های عمومی توابع kernel را داشته باشد. در این حالت مرکز کلاسترها به صورت صریح محاسبه نمی‌شود و فاصله هر نقطه از مرکز کلاستر به صورت ضمنی محاسبه می‌شود. الگوریتم کلی به این صورت است[1]:

Algorithm outline: Kernel *k*-Means

Input: Kernel matrix K, Number of clusters k, Initial clusters $C_1, \dots, C_k$

Output: Final clusters $C_1, \dots, C_k$, Clustering error E

1. For each point $\boldsymbol{x}_n$ and every cluster C_i compute $\|\phi(\boldsymbol{x}_n) - \boldsymbol{m}_i\|^2$ using (3)
2. Find $c^*(\boldsymbol{x}_n) = \text{argmin}_i(\|\phi(\boldsymbol{x}_n) - \boldsymbol{m}_i\|^2)$
3. Update clusters as $C_i = \{\boldsymbol{x}_n | c^*(\boldsymbol{x}_n) = i\}$
4. If not converged go to step 1 otherwise stop and return final clusters $C_1, \dots, C_k$ and E calculated using (2).

که در مرحله یک الگوریتم، از رابطه زیر برای محاسبه فاصله هر داده از مرکز کلاسترها استفاده می‌شود:

$$\|\phi(\boldsymbol{x}_i) - \boldsymbol{m}_k\|^2 = K_{ii} - \frac{2\sum_{j=1}^{N} I(\boldsymbol{x}_j \in C_k) K_{ij}}{\sum_{j=1}^{N} I(\boldsymbol{x}_j \in C_k)} + \frac{\sum_{j=1}^{N}\sum_{l=1}^{N} I(\boldsymbol{x}_j \in C_k) I(\boldsymbol{x}_l \in C_k) K_{jl}}{\sum_{j=1}^{N}\sum_{l=1}^{N} I(\boldsymbol{x}_j \in C_k) I(\boldsymbol{x}_l \in C_k)} \quad (3)$$

d) تابع هزینه که همان واریانس داده‌ها حول مرکز داده‌هاست به این صورت تعریف می‌شود:

$$j = \sum_{c \in clusters} \sum_{x \in c} ||x - \mu_c||^2$$

طبق این رابطه با افزایش تعداد کلاسترها، فاصله هر داده نسبت به مرکز کلاستر مربوط به آن داده کاهش می‌یابد و در نتیجه مجموع واریانس‌ها کاهش می‌یابد.

e) با توجه به فرمولی که در بخش d ارائه شد، اگر تعداد کلاسترها با تعداد داده‌ها برابر باشد، مجموع واریانس صفر خواهد شد. زیرا در این حالت هر داده در کلاستری قرار می‌گیرد که مرکز آن خودش است و درنتیجه فاصله آن از مرکز کلاستر صفر خواهد بود.

[1] The Global Kernel k-Means Clustering Algorithm/ Grigorios Tzortzis and Aristidis Likas/ International Joint Conference on Neural Networks/ 1-8 June 2008