سید محمد طاها طباطبایی-9812762838 -گزارش تمرین شماره 1

برای بخش اول و اجرای الگوریتم های خوشه بندی، از لایبرری sklearn استفاده کردم. این لایبرری شامل پیاده‌سازی انواع مختلفی از الگوریتم های خوشه بندی است. سه الگوریتم kMeans,DBSCAN,Agglomerative در این لایبرری موجود بود. پارت اول کدم یعنی از ابتدا تا خط 37، مربوط به خوندن فایل عکس‌ها و تبدیل اونها به آرایه میشه، که هر عکس رو با تابع

matplotlib.image.imread()

میخونم، خروجی رو که 80*70 هست با تابع

numpy.ndarray.flatten()

تبدیل به آرایه یک بعدی با طول 5600 میکنم.در انتها هم عکس رو به آرایه imageArr اضافه میکنم.

لیبل های اصلی رو هم توی آرایه myLabels مینویسم.

تو خط 41 کلاس Kmeans رو ایمپورت میکنم. متد fit توی کلاس‌های لایبرری sklearn، خوشه بندی رو اجرا میکنه.

پارامتر ورودی init=random مشخص میکنه که مراکز خوشه ها به صورت رندوم انتخاب بشن.

پارامتر n_clusters=40 تعداد خوشه ها رو برابر 40 ست میکنه.

```
kmeans = KMeans(init="random",n_clusters=40).fit(imageArr)
```

خط 47 و 49 برای خوشه بندی DBSCAN که پارامتر eps=2500، همون مشخصه اپسیلون رو تعیین میکنه و min_samples=3 هم تعداد حداقل داده توی خوشه برای اینکه نویز محسوب نشه .البته مقادیر اپسیلون و حداقل تعداد داده خوشه رو با سعی و خطا به دست آوردم(در مرحله اول) و ممکنه بشه خروجی کمی بهتری هم گرفت(مقادیری که اینجا گذاشتم با توجه به خروجی پارت بعدی تعیین شده) .

```
dbscan = DBSCAN(eps=2500,min_samples=3).fit(imageArr)
```

خط 53 تا 57 هم برای خوشه بندی Agglomerative.

پارامتر linkage که سه حالت average,single,complete داره، سه حالت خوشه بندی رو مشخص میکنه. n_clusters=40 هم تعداد خوشه هایی که الگوریتم اگر به این تعداد رسید متوقف بشه.

```
aglo_avg = AgglomerativeClustering(linkage='average',n_clusters=40).fit(imageArr)
aglo_single = AgglomerativeClustering(linkage='single',n_clusters=40).fit(imageArr)
aglo_comp = AgglomerativeClustering(linkage='complete',n_clusters=40).fit(imageArr)
```

از خط 63 تا انتهای هم خروجی خوشه بندی ها به الگوریتم rand Index که توی فایل Rand_Index.py نوشتم داده میشه و مقادیر RI چاپ میشه.

فایل Rand_Index.py شامل متد rand_index میشه که پیاده‌سازی فرمول RI هستش.این متد labels_true که لیبل های اصلی هستن و labels_predicted که لیبل های خروجی الگوریتم هامون هستن رو به عنوان ورودی میگیره. توی فور تو در تو، لیبل های جدید با هم جفت میشن و با لیبل های اصلی مقایسه میشن و بر اساس 4 حالت ممکن، مقدار یکی از 4 متغییر trueP, trueN, falseP یا falseN زیاد میشه.

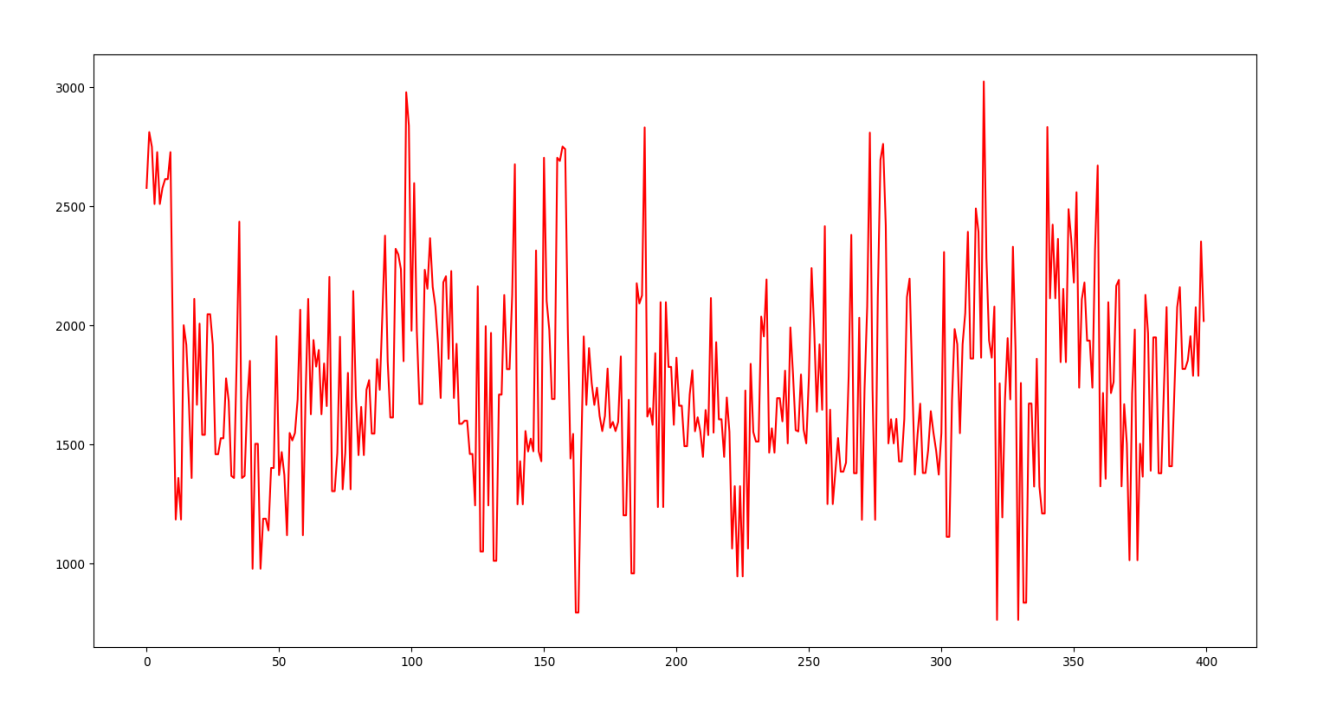
در نهایت هم مقدار RI محاسبه میشه و ریترن میشه.

نتایج خروجی خوشه بندی ها به این شکله:

```
PS C:\Users\siedt> & C:/Users/siedt/AppData/Local/Programs/Python/Python310/python.exe e:/taha/code/HW1/src/phase1.py

*************
KMeans(n_clusters=40):
trueP: 1109 trueN: 76784 falseP: 1216 falseN: 691
RI: 0.9761027568922306

*************
DBSCAN(eps=2500,min_samples=3):
trueP: 1441 trueN: 65841 falseP: 12159 falseN: 359
RI: 0.8431328320802005

*************
AgglomerativeClustering(linkage='average',n_clusters=40):
trueP: 1383 trueN: 74840 falseP: 3160 falseN: 417
RI: 0.9551754385964912

*************
AgglomerativeClustering(linkage='single',n_clusters=40):
trueP: 1468 trueN: 48160 falseP: 29840 falseN: 332
RI: 0.621904761904762

*************
AgglomerativeClustering(linkage='complete',n_clusters=40):
trueP: 1194 trueN: 76509 falseP: 1491 falseN: 606
RI: 0.9737218045112782
PS C:\Users\siedt>
```

برای فاز دوم، ایده بهبود من مربوط میشه به اینکه برای الگوریتم DBSCAN، روشی ارائه بدیم که انتخاب مقدار اپسیلون به جای سعی و خطا، طبق یک قاعده ریاضی و بر اساس داده‌های دیتاست مشخص بشه تا با بررسی داده‌های دیتاست، بتونیم بهترین اپسیلون ممکن رو پیدا کنیم

روشی که مدنظر من هست به این شکله که برای داده‌های دیتاست، میانگین فاصله این داده از k داده نزدیکتر بهش محاسبه بشه. این میانگین احتمالاً بتونه تقریب خوبی برای مقدار اپسیلون باشه، البته مهمه که مقدار k چند باشه. بعد از اینکه برای همه داده(یا برای یه بخشی از داده توی مسائل بزرگتر) میانگین فاصله های رو حساب کردیم، می تونیم از این داده‌ها برای انتخاب اپسیلون بهتر استفاده کنیم(خط 80 تا انتهای کد). نمایش میانگین ها با k=5 به صورت مصور به شکل زیر:



اینجا به نظر میاد که چون تقریباً میانگین فاصله اکثر نقاط تا حداکثر کمتر از 2500 هست، احتمالاً انتخاب این عدد برای اپسیلون فاصله مناسبی باشه. این ایده رو از این لینک و این مطلب در سایت مدیوم و این مقاله گرفتم و استفاده کردم.