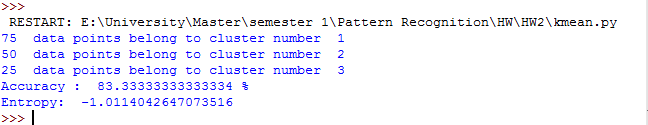
برای حل این سوال از زبان برنامه نویسی پایتون استفاده شده است.

* الگوریتم Kmeans

برای پیاده سازی الگوریتم kmeans ابتدا 2 کلاس DataPoint و Centroid تعریف میکنیم. کلاس اول برای نقاط 4 بعدی داده شده و کلاس دوم برای نقاط مرکزی اولیه است. سپس داده های موجود در فایل iris.txt را با استفاده از تابع load\_data میخوانیم و نقاط را میسازیم و درون یک آرایه میریزیم. سپس 3 نقطه اولیه را به صورت رندم میسازیم و به درون آرایه centroids میریزیم.

در هر مرحله از تابع calculate\_centroids() فاصله هر نقطه رو تا هر 3 مرکز محسابه کرده و نزدیک ترین را می یابیم و به همان کلاستر اختصاص میدهیم.

پس از 100 بار اجرای این الگوریتم نتیجه ای مانند شکل زیر بدست می آید:



نتیجه بالا میانگین صدبار اجرای الگوریتم است.

اجرای االگوریتم به تعداد 100بار حدودا 3 دقیقه طول کشید.

* الگوریتم Agglomerative Hierarchical

در این الگوریتم ابتدا مانند سوال قبل داده هارا در کلاسی به نام DataPoint ذخیره میکنیم. سپس هر نقطه را در کلاستر مخصوص به خود قرار میدهیم. فقط نکته اینجاست که برای عملکرد بهتر مسئله ابتدا داده های تکراری را در فاز preprocess حذف میکنیم.

نتیجه نهایی اجرای دو الگوریتم را در جدول زیر میبینیم:

|  |  |  |  |
| --- | --- | --- | --- |
| نام الگوریتم | مدت زمان اجرا | میانگین دقت | entropy |
| Kmeans | 173 s | 83.33 | -1.01 |
| Agglomerative Hierarchical | 14.3 | 85.6 | -0.9333 |

توضیح توابع فایل kmeans.py

|  |  |
| --- | --- |
| نام تابع | توضیح |
| Load\_data | خواندن از فایل ورودی و برگرداندن یک تاپل (ورودی،خروجی) |
| initialize\_centroids | مقدار دهی اولیه K نقطه مرکزی به صورت رندوم |
| get\_distance | فاصله اقلدیسی 2 نقطه |
| Calculate\_centroids | آپدیت نقاط اولیه با ورود هر نقطه جدید از ورودی طبق تعریف الگوریتم |
| Show\_centroids | نمایش نقاط مرکزی |
| show\_results | نمایش تعداد داده های کلاستربندی شده در هر کلاستر |
| calculate\_entropy | محاسبه entropy |

توضیح توابع فایل agnes.py

|  |  |
| --- | --- |
| نام تابع | توضیح |
| Load\_data | خواندن از فایل ورودی و برگرداندن یک تاپل (ورودی،خروجی) |
| init\_clusters | اختصاص هر نقطه به یک کلاستر |
| get\_distance | فاصله اقلدیسی 2 نقطه |
| get\_sinle\_linkage\_distance | محاسبه فاصله single linkage (نزدیک ترین فاصله بین نزدیک ترین نقاط دو کلاستر) |
| show\_cluster | نمایش داده های کلاستربندی شده در هر کلاستر |
| update\_clusters | ترکیب نزدیک ترین 2 کلاستر با هم |