گزارش فاز اول پروژه هوش محاسباتی

محمدرضا اسماعیلیان ۹۹۱۲۷۶۲۵۷۴ – حسین آقاسی زاده ۹۹۱۲۷۶۲۳۶۹

- **بخش اول خوشه بندی:**

در بخش اول ما 5000 داده اول فایل آموزش را انتخاب و روش های مختلف خوشه بندی داده ها را بر روی آنها اعمال میکنیم که کد آن در فایل Finding.Best.Algorithm.py آمده است. در اینجا ما بدین صورت عمل میکنیم که الگوریتم K-Means و DB-Scan را بر روی این داده ها اعمال میکنیم و در نهایت نتیجه آنها را با تبدیل 2048 بعد به 2 بر روی صفحه نمایش می دهیم. اما مشکل اصلی در این بخش این نبوده بلکه مشکل اصلی که ما با آن مواجه هستیم انتخاب پارامتر برای این الگوریتم هاست و اینکه به طور کلی کدام یک از این الگوریتم ها مناسب می باشد؟

راه حل ما پلات کردن داده های خوشه بندی شده برای همه مقادیر پارامترهاست، اما چگونه؟ ما به الگوریتم K-Means تمامی مقادیر بین 4 تا 12 و در الگوریتم DB-Scan مقدار اپسیلون را 0.001 در نظر گرفته و در هر مرحله آن را ده برابر میکنیم تا حدود مناسب ترین پارامتر یافت شود.

نتایج بدست آمده (قابل مشاهده در پوشه Finding.Best.Algorithm) از مرحله قبل حاکی از آن است که حداقل و بهترین مقدار k برای الگوریتم اول 5 و هر مقدار کمتر از آن اور کلاسترینگ یا به عبارتی خوشه بندی بیش از نیاز است که بسته به شرایط میتواند مفید یا مضر باشد که در بخش ها آتی به ترتیب از مقداری بیشتر از 5 و سپس از 5 به عنوان شاخص خوشه بندی استفاده می کنیم. همچنین این نتایج به ما نشان داد الگوریتم DB-Scan با هیچ پارامتری به عنوان اپسیلون نمیتواند مفید باشد و تمامی داده ها را به یک خوشه تقسیم بندی خواهد کرد یا به قدری سایر خوشه ها را کوچک خواهد کرد که باعث بی ارزش شدن آن میشود.

حال در مرحله دوم که ما حدود و الگوریتم مناسب خود را پیدا کردیم سراغ روش های دقیق تر ارزیابی میرویم که در فایل Accurate.Analysis.py روش اینکار آمده است. در این مرحله نه تنها داده ها را بر روی دو بعد رسم میکنیم بلکه آنها را بر روی سه بعد برای آنالیز دقیق تر منتقل میکنیم. همچنین از متریک های موجود در کتابخانه sklearn برای ارزیابی دقیق تر نیز استفاده میکنیم. که عبارتند از:

Silhouette score:

این معیار برای ارزیابی کیفیت خوشه های ایجاد شده با استفاده از الگوریتم های خوشه بندی از نظر میزان خوشه بندی نمونه ها با نمونه های مشابه دیگر استفاده میشود. مقدار امتیاز از **1-** تا **1+** متغییر است. اگر امتیاز یک باشد، خوشه متراکم و به خوبی از سایر خوشه ها جدا شده است. مقدار نزدیک به صفر نشان دهنده خوشه های همپوشانی با نمونه های بسیار نزدیک به مرز تصمیم گیری خوشه های همسایه است. امتیاز منفی نماین کننده این است که امکان دارد نمونه ها با خوشه اشتباه قرار گرفته اند.

Calinski harabasz score:

این پارامتر که با عناون معیار نسبت واریانس شناخته میشود به عنوان نسبتی از مجموع پراکندگی بین خوشه ای و مجموع پراکندگی درون خوشه ای برای همه خوشه ها محاسبه می شود.

Davies bouldin score:

این امتیاز به عنوان میانگین اندازه گیری شباهت هر خوشه با شبیه ترین خوشه اش تعریف می شود، که در آن شباهت نسبت فاصله های درون خوشه ای به فاصله های بین خوشه ای است. بنابراین، خوشه هایی که دورتر از هم هستند و کمتر پراکنده هستند، امتیاز بهتری بدست می آورند. که در اینجا منظور از امتیار بهتر امتیاز کمتر است.

نتایج این مرحله را نیز می توانید در پوشه Accurate.Analysis مشاهده کنید. همانطور که از نمودار های سه بعدی و نتایج پارامتر ها نمایان است بازهم میتوان متوجه شده که 5 بهترین پارامتر برای الگوریتم K-Means بوده و قابل مشاهده است که دیتای مارا در پنج محدوده قابل مشاهده در صفحه دو بعدی به خوبی تقسیم و مقادیر بسیار مناسبی را از پارامتر ها نتیجه خواهد داد.

- **بخش دوم پیدا کردن تعداد دامین:**

حال سوال این است که آیا پنج خوشه کافیست؟ جواب ما در پاسخ به این سوال مثبت بوده زیرا که اورکلاسترینگ میتواند در این مسئله که داده ها هم برچسب و هم دامنه دارند آسیب زده و این امر باعث میشود خوشه بندی های با بیشتر از پنج داده هایی که باید در دو دامین مختلف باشند را باهم ترکیب کند و حل مسئله را دچار اختشاش کند. در این موقعیت لازم میبینم که به این مسئله اشاره کنم که از آنجا که پلات خروجی با استفاده از **tsne** چون پنج ناحیه دارد نتیجه ما نیز نهایتا باید به پنج دامنه برسد.

در اینجا برای شفاف سازی بخش ترکیب خوشه هارا با استفاده از پارامتر **k** ی **10** حل کرده تا نمایش دهیم درصورت مواجه در سایر مسائل با چنین مشکلی چگونه خوشه هارا باهم ادغام کنیم. برای این کار دو راه حل وجود دارد که هر دو شامل خوشه بندی **Agglomerative** می باشند.

راه حل اول آن است که داده ها را به خوشه های بیشتر تقسیم کنیم اینگونه که این الگوریتم را برای داده ها اجرا و در نهایت نتیجه این الگوریتم و الگوریتم قبلی را به عنوان یک عدد دو رقمی گزارش دهیم که نمایش نهایی آن در پوشه **Combining.Clusters** آمده است که قابل مشاهد است.

همچنین می توان از مراکز خوشه ها برای پارامتر ورودی این تابع نیز استفاده کرد. اما سوال اینجاست که چرا روش دوم به ظاهر ضعیف تر از روش اول است پاسخ در این است که با افزایش مراکز خوشه های ورودی دقت الگوریتم دوم بالاتر و بالاتر می رود زیرا خوشه ها بیشتر از هم تفکیک شده و هنگام ترکیب بین دو خوشه قرار نمیگیرند. پیاده سازی این الگوریتم را میتوانید در فایل **Combining.Clusters.py** مشاهده کنید.

- **بخش سوم ارزیابی با برچسب های دامین:**

در این مرحله ما ارزیابی خود را به دو روش انجام می دهیم که هر دو روش در فایل **Testing.py** قابل مشاهده هستند و همچنین میتوان خروجی پلات را نیز در پوشه اصلی مشاهده کنید. روش اول **Accuracy score** میباشد که در صورت برگرداندن مقداری مثبت برای یک مجموعه از تست کیس

ها شما خوشه بندی مناسبی دارید. این تابع پیشبینی شما از خوشه های یک دیتاست آزمایشی و لیبل درصت آنها را میگیرد که در مورد این سوال نتایج پیشبینی همان نتیجه ما برای تعیین دامین و لیبل دامین واقعی داده ها در فایل است.

روش دوم تابعی میباشد که براساس نحوه کارکرد الگوریتم خود نوشته ایم. در این تابع ما با توجه به اینکه داده هایی که در یک دامین در پیشبینی ما هستند باید در دامین های واقعی نیز در یک دامین باشند به آنها امتیاز می دهیم که نتیجه آن 77% شباهت بین خروجی ما با دامین واقعی است که با توجه به نقصان های الگوریتم شباهت سنجی ما مقدار قابل قبولی است.