دانشگاه صنعتی امیرکبیر
دانشکده مهندسی کامپیوتر

مبانی هوش محاسباتی
تمرین پیاده‌سازی فازی
(Fuzzy C-Means Clustering)

استاد درس: دکتر عبادزاده

بهار ۱۴۰۰

مقدمه

توی درس با منطق فازی آشنا شدید و مباحث تئوری‌اش رو خوندید. ایده‌های کلی قضیه این بود که بیایم مجموعه‌هایی تعریف کنیم که عناصر مختلف بتونن به درجات مختلف عضو اون مجموعه‌ها باشن، گزاره‌هایی تعریف کنیم که بتونن به طور نسبی درست یا غلط باشن، بتونیم پلی ایجاد کنیم بین قوانین منطقی و محاسبات ریاضی به طوری که این قوانین بتونن در درجات مختلفی صادق باشن و مواردی از این قبیل.

بعد از جلو رفتن مباحث تئوری این حوزه، خیلی‌ها شروع کردن این ایده‌ها رو ببرن توی سایر بخش‌های علوم کامپیوتر و به الگوریتم‌ها و مسائل مختلف، از این منظر نگاه کنن. حاصلش میشه حل مسائل Classification به کمک الگوریتم‌های مبتنی بر منطق فازی، تعریف شدن شبکه‌های عصبی-فازی که بر طبق قوانین منطقی فازی کار می‌کنن و...

توی این تمرین پیاده‌سازی قراره که سراغ الگوریتم خوشه‌بندی K-Means[1] بریم و نسخه‌ی فازی اون یعنی Fuzzy C-Means رو پیاده‌سازی کنیم. یکی از بهبودهایی که این الگوریتم نسبت به حالت غیرفازی‌اش داره اینه که داده‌ها فورس نمی‌شن که فقط به یک خوشه‌ی خاص تعلق پیدا کنن بلکه می‌تونن به درجات مختلف به خوشه‌ها تعلق پیدا کنن (این مورد برای نقاطی که در حالت مرزی بین خوشه‌ها قرار دارن، اتفاق می‌افته). در نتیجه، خوشه‌بندی می‌تونه به صورت منعطف‌تری انجام شه.

---

[1] الگوریتم K-Means رو می‌تونید از این لینک مرور کنید.

الگوریتم Fuzzy C-Means

در قدم اول، توی این الگوریتم، مثل K-Means تعداد خوشه‌هایی که می‌خوایم رو مشخص می‌کنیم. بعد به اون تعداد، مرکز خوشه (Centroid) اولیه به صورت رندوم تولید می‌کنیم.
در ادامه، توی یک حلقه باید دوتا کار رو انجام بدیم:
۱- پیدا کردن اینکه هر داده به کدوم خوشه (یا خوشه‌ها) تعلق داره.
۲- آپدیت کردن مرکز خوشه‌ها براساس داده‌های متعلق بهشون.

برای کار اول، توی C-Means به هر خوشه به چشم یک مجموعه فازی نگاه میشه. در نتیجه، هر داده به تمام خوشه‌ها تعلق داره ولی به اندازه‌های مختلف. میزان تعلق داده‌ی k ام به خوشه‌ی i ام از رابطه‌ی زیر محاسبه میشه:

$$u_{ik} = \frac{1}{\sum_{j=1}^{c} \left(\frac{||X_k - V_i||}{||X_k - V_j||} \right)^{\frac{2}{m-1}}}$$

X_k داده‌ی k ام، c تعداد خوشه‌ها، V_i مرکز خوشه‌ی i ام و m یک پارامتر (بزرگ‌تر از ۱) عه که باید برای الگوریتم مشخص کنیم. کاری که رابطه‌ی بالا انجام میده اینه که میزان تعلق داده به یک خوشه رو با توجه به نزدیکی به مرکز اون خوشه و مقایسه‌اش با نزدیکی به سایر مراکز خوشه‌ها حساب می‌کنه.

برای کار دوم، باید میانگین نقاطی که به خوشه تعلق دارن رو حساب کنیم و اون رو به عنوان مرکز خوشه‌ی جدید در نظر بگیریم. از اونجا که همه‌ی نقاط عملا عضو تمامی خوشه‌ها هستن، پس باید میانگین وزن‌دار بگیریم.

$$V_i = \frac{\sum_{k=1}^{N} u_{ik}^m X_k}{\sum_{k=1}^{N} u_{ik}^m}$$

اینطوری، اون داده‌هایی که تعلق بیشتری دارن به یک خوشه، بیشتر نقش دارن توی تعیین مرکز اون خوشه؛ که منطقی هم هست.

پس بدین شکل این دوتا کار رو توی یک حلقه باید انجام بدیم. این حلقه رو مثلا به ازای ۱۰۰ بار اجرا می‌کنیم که مطمئن بشیم خوشه‌ها به ثبات رسیدن و دیگه تغییر زیادی نمی‌کنن.

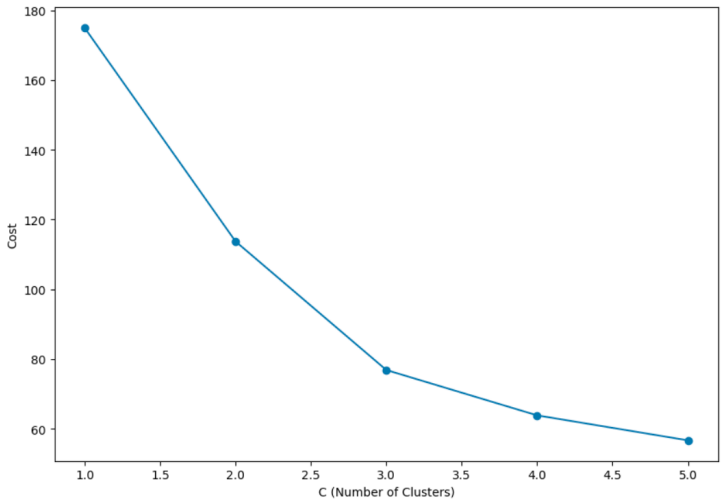
تابع هزینه و نحوه انتخاب C بهینه

توی قسمت قبل گفتیم که حلقه‌ی مربوط به الگوریتم رو باید تا جایی ادامه بدیم که خوشه‌ها به ثبات برسن. سوالی که مطرحه اینه که به ثبات رسیدن یعنی چی و اصلا چرا باید این اتفاق بیافته؟ نکته اینجاست که دوتا کاری که داریم توی حلقه انجام میدیم، توی پشت‌پرده دارن تابع هزینه زیر رو Minimize می‌کنن:

$$J = \sum_{j=1}^{N} \sum_{i=1}^{c} u_{ij}^{m} \ ||X_j - V_i||^2$$

یعنی دارن فاصله‌ی هر داده از مراکزی که بهش تعلق داره رو حداقل می‌کنن (در واقع با مشتق گرفتن از این تابع و برابر صفر قرار دادنش میشه رسید به همون دوتا فرمول قبل).
پس ما داریم توی خوشه‌بندی با C-Means، یک مسئله‌ی بهینه‌سازی حل می‌کنیم و به صورت Iterative پیش میریم تا به مقدار حداقلی برسیم.

نکته‌ی بعدی اینه که تاثیر تعداد خوشه‌ها بر روی تابع هزینه چیه؟ اگر به ازای C های مختلف الگوریتم رو اجرا و این مقدار رو پلات کنیم، یک چنین نموداری خواهیم داشت:



همانطور که مشاهده می‌کنیم، با افزایش تعداد خوشه‌ها، Cost سیر نزولی داره. منطقی هم هست، چون وقتی تعداد مرکز خوشه‌ها زیاد میشه، فاصله‌ی داده‌ها هم از این مراکز کاهش پیدا می‌کنه.

حالا سوالی که مطرحه اینه که چه C ای رو انتخاب کنیم؟ خیلی راه ایده‌آلی برای انتخاب C بهینه وجود نداره؛ اما چندتا روش هست که معمولا از اون‌ها استفاده میشه.
یکی از اون‌ها، روش Elbow هست که میگه همون نمودار بالا رو پلات کنیم و اون C ای رو انتخاب کنیم که از اونجا به بعد، دیگه خیلی هزینه کاهش چشمگیری پیدا نکنه[2]. در نتیجه با توجه به شکل بالا، C=3 می‌تونه گزینه مناسبی باشه.

[2] توضیحات بیشتر: لینک

موارد تحویلی

الگوریتم C-Means رو پیاده‌سازی کنید و بر روی ۴ دیتاستی که در اختیار دارید اجرا کنید.

- برای هر دیتاست، نمودار هزینه برحسب C رو پلات کرده و از طریق روش Elbow، تعداد خوشه‌های بهینه رو تعیین کنید.
- برای دیتاست اول، چند مقدار مختلف برای m را امتحان کرده و در مورد تاثیر آن بر روی خوشه‌های ایجاد شده و نمودار هزینه بحث کنید.

از اونجایی که داده‌ها به همه‌ی خوشه‌ها تعلق دارن، در نتیجه نمی‌تونیم خیلی ساده بیایم برای هر خوشه یه رنگ خاص در نظر بگیریم و داده‌های متعلق به اون خوشه رو با اون رنگ نشون بدیم و پلات کنیم (در واقع برای نمایش دقیق خروجی خوشه‌بندی به صورت فازی، نیازه که یه Color Gradient طوری رو در نظر بگیریم). در نتیجه برای نمایش ساده خروجی، میایم خوشه‌ها و داده‌ها رو به دنیای Crisp برمی‌گردونیم. کافیه که ببینیم هر داده به کدوم خوشه بیشتر از بقیه‌ی خوشه‌ها تعلق داره، و از اون طریق رنگ اون خوشه رو بهش بدیم.

- برای داده‌های دوبعدی دیتاست، این کار رو انجام بدید و پلات کنید (مراکز خوشه‌ها را نیز پلات کنید.)