امیرحسین قاسمی لمراسکی - ۹۹۲۵۰۷۱ - دکتر زرین بال ماسوله

My github: amirgh2001

githubRepo for this exam: link

سوال ۱:

در این سوال ابتدا تابعی برای k-means clustering مینویسیم و سپس تابع را تست میکنیم با یک نمونه. سپس ورودی را با فرمت خواسته شده از کاربر میگیریم و با تابع مربوطه آن را خوشه بندی میکنیم. در نهایت در بخش آخر زمان بر ترین حالت ها را تست میکنیم که زمان انجام این خوشه بندی بیشتر از ۳ ثانیه نشود.

در سلول اول، کتابخانه های مربوطه را ایمپورت میکنیم. Random برای ساختن نقاط تصادفی و بخشی از تابع k-means . Math برای کار های ریاضیاتی. Matplotlib برای تست تابع kmeans و visualise کردن نقاط خوشه بندی شده.

در سلول دوم تابعی به نام distance را نوشته ام که فاصله اقلیدسی میان دو نقطه به مختصات (X1, Y1) و (X2, Y2) را محاسبه میکند. سپس با یک نمونه تست کردم تابع را که مطمین شوم تابع را درست نوشته ام.

در سلول سوم تابعی به نام centroids را تعریف میکنیم که لیستی از تاپل ها را میگیرد که هر تاپل شامل x و y نقاط ورودی است.سپس با یک نمونه این تابع را تست گرفتم که درست کار کند.

در سلول چهارم تابه اصلی که تابع kmeans است را نوشتم که لیستی از توپل ها که شامل همه نقاط است را میگیرد و k که تعداد دسته هایی که میخواهیم به آن تعداد خوشه بندی انجام شود و max iteration که تعداد ماکسیمم دفعاتی ست که محاسباتمان و آپدیت کردن نقاط ورودی را انجام میدهیم. ابتدا مرکز اولیه را با استفاده از random.sample انتخاب میکنیم.
سپس به طور مکرر مکان هر مرکز را تا زمانی که حداکثر تعداد تکرارها به دست آید، به روز میکنیم. در هر تکرار، تابع فهرستی از k لیست خالی برای نمایش خوشه ها ایجاد می کند و با محاسبه فاصله اقلیدسی بین نقطه و هر مرکز با استفاده از تابع فاصله، هر نقطه در لیست نقاط ورودی را به نزدیکترین مرکز به مرکز اختصاص می دهد(فرایند کلاسترینگ). سپس نقطه به لیستی که نشان دهنده خوشه ای است که به آن تعلق دارد اضافه می شود. سپس در نهایت لیستی از نقاط را به صورت دستی به تابع میدهیم تا ببینیم تابعمان درست کلاستر میکند یا نه.

در سلول بعدی (سلول پنجم) خروجی خوشه بندی شده سلول قبلی را با استفاده از matplotlib روی نمودار میآوریم تا بصورت بصری ببینیم این خوشه بندی درست انجام شده یا خیر. رنگ خوشه ها را هم به ترتیب قرمز، سبز و آبی میگذاریم.

در سلول ششم، تابعی به نام find_closest_points نوشتیم که به عنوان ورودی مجموعه خوشه هایی که شامل نقاط هستند را میگیرد و به عنوان خروجی به ما کوتاه ترین فاصله میان دو نقطه از دو خوشه مختلف را میدهد.
ابتدا یک دیکشنری خالی تعریف میکنیم که قرار است فاصله اقلیدسی میان جفت نقاط را برای ما ذخیره کند. سپس در خط بعد یک حلقه for داریم که روی هر خوشه در لیست خوشه ها تکرار میشود و ایندکس هر خوشه را به i و خود خوشه را به ، cluster_i assign میکند.
سپس در خط بعدی (حلقه for توی حلقه اول) این حلقه for روی هر خوشه در لیست خوشه‌ها که بعد از cluster_i قرار می‌گیرد تکرار می‌شود و ایندکس خوشه را به j و خود خوشه را به assign ، cluster_j میکند. با اینکار مطمین میشویم که هر جفت خوشه فقط یک بار با هم مقایشه میکنیم. در شش خط بعدی فاصله اقلیدسی نقاط خوشه های مختلف (cluster_i , cluster_j) را دو به دو محاسبه و در دیکشنری که ساخته ایم ذخیره میکنیم. از شرط if بدین منظور استفاده کردم که فاصله نقاطی که قبلا محاسبه شده اند را مجددا در دیشکنری اد نکنیم. و با استفاده از round نیز فاصله تا ۷ رقم اعشار گرد میکنیم.
سپس در مرحله بعد دیکشنری ساخته شده را به ترتیب صعودی بر اساس value ها، sort میکنیم و کمترین فاصله را return میکنیم. و تست تابع هم که در اخر هر سلول انجام میشود :)

در سلول هفتم داده ها را از کاربر با توجه به فرم خواسته شده دریافت میکنیم. x,y نقاط را به صورت لیستی از توپل ها ذخیره میکنیم و k را نیز از کاربر میگیریم که بدانیم به چند خوشه باید تقسیم کنیم.

در سلول هشتم، از کتابخانه time استفاده کردم تا زمان ران شدن کد را پیدا کنم سپس ورودی های سلول قبل را به تابع های مربوطه دادم تا آنها را خوشه بندی کند و فاصله نزدیک ترین نقاط از دو خوشه مختلف را پیدا کند. سپس زمان پس از ران شدن کد و زمان بعد از ران شدن کد را از هم کم کردم تا ببینیم ران شدن کد چقدر زمان میبرد (نمونه سمپلیست که در صورت سوال داده شده بود).

در سلول هشتم لیستی از ۲۰۰ نقطه که مختصات x, y همگی بین -۱۰۰ و +۱۰۰ و اعشاری هستند را ساختم بزای اینکه ببنیم آیا کدمان میتواند در کمتر از ۳ ثانیه بدترین و سخت ترین حالت ممکن را انجام دهد یا خیر.

در تمامی سلول های بعدی، این دویست نقطه را ابتدا به ۱۹۹ خوشه. سپس ۱۰۰، ۵۰، ۲۵، ۱۲، ۶، ۳ خوشه تقسیم کردم و زمان انجام هرکدام را پرینت کردیم به منظور براورده کردن خواسته مساله. مشاهده میکنیم که هر چه تعداد کلاستر ها بیشتر باشد این کار زمانبر تر هست و برای بدترین حالت ممکن که ۱۹۹ خوشه هست، ۲ ثانیه زمان نیاز هست و همینطور زمان انجام برای K های کمتر، کمتر میشود تا اینکه برای k=3 زمان انجام کار به ۷ صدم پانیه میرسد. مشاهده میشود که در همه حالات زمان انجام کمتر از ۳ ثانیه.

سوال ۲

ابتدا یک دیکشنری میسازیم و در آن مشخص میکنیم که هر edge به چه edge های دیگری راه دارد. سپس از روی این داده ای که ساختیم همه ی مسیر های ممکن در این گراف را پیدا میکنیم. (از یک خانه شروع و به خانه های دیگر رفته و نهایتا به همان خانه ختم میشود)
حال ما همه سیکل هارو که داریم. اینارو فیلتر میکنیم تا فقط اونایی که مساله ازمون خواسته حتما ازونا رد بشن رو پیدا کنیم ونود شروع که نود پایان نیز هست هم نود اول سیکلمون باشه. حال اگر چند مسیر بودند که این شرایط را دارا بودن اونی که کمترین هزینه رو داره به عنوان خروجی بدیم.

در این سوال وقت نشد که کل الگوریتمم را تبدیل به کد کنم و فقط تا مرحله قبل از فبلتر کردن را انجام دادم.

در سلول اول یک دیکشنری میسازیم که در آن key ها به ترتیب شماره نود ها هستن و value ها نود هایی هستند که به آن نود راه دارند.
در سلول بعد یک تابع نوشتم که همه ی مسیر های ممکن که از یک سلول شروع میکند و به همان سلول ختم میشود را نشان میدهد. با یک استک که شامل گره شروع و یک مسیر خالی است شروع میکنم که نشان دهنده مسیر اولیه ایست که میخواهیم آن را بررسی کنیم. سپس آن نود را حذف میکنیم. سپس چک میکنیم که آیا مسیر فعلی به نود پایانی منتهی میشود یا خیر. اگر میشود، ادامه میدهیم. در کل نود های همسایه iterate میکنیم و برای هر نود همسایه، اگر در مسیر فعلی وجود داشت، skip میکنیم. در نهایت مسیر ها را در یک لیست برمیگردانیم.

سوال ۳:

تابع longest_substring در رشته را به صورت ورودی میگیرد و طولانی ترین زیررشته آنان را پیدا میکند و برمیگرداند.
فرمت خروجی همانطور که سوال میخواهد،به صورت یک تاپل هست که ۳ عدد دارد. عدد اول ایندکس زیررشته مسترک در استرینگ اول. عدد دوم اینکس اول زیررشته مشترک در رشته دوم و عدد سوم نیز طول بلند ترین زیر رشته مشترک است.
ابتدا طول رشته های ورودی را در m, n ذخیره میکنیم.
سپس لیست dp را تعریف میکنیم. لیست دوبعدی dp با اندازه (m+1) (n+1) را با تمام ورودی‌ها مقدار دهی میکنیم. این لیست برای ذخیره طول طولانی‌ترین زیررشته مشترک بین str1 و str2 استفاده می‌شود که با کاراکتر i ام از str1 و با کاراکتر jام از str2 تمام میشود.

سپس دو متغیر برای ذخیره کردن طول بزرگترین زیررشته مشترک و ایندکس آخرین کارکتر در str1 که آخرین بخش بلند ترین زیررشته مشترک است را میسازیم و مقدار اولیه شان را ۰ میگذاریم.
سپس حلقه های تو در تو شروع میشود! دو حلقه تو در تو یکی برای رشته اول و یکی برای رشته دوم مینویسم.
خط بعد یک شرط if است. (if str1[i-1] == str2[j-1]) این شرط چک میکند که اگر کارکتر i ام از str1 برابر با کارکتر j ام از str2 برابر هست یا خیر. اگر برابر باشد یعنی اینکه یک زیررشته مشترک به طول dp[i-1][j-1] + 1 که در کارکتر i از رشته اول و j از رشته دوم پایان میابد، وجود دارد. این مقدار را در dp[i][j] ذخیره میکنیم.
در خط بعدی چک میکنیم که طول زیررشته مشترک که در ایندکس i از رشته اول و j از رشته دوم پایان میابد، بزرگتر از max_length بزرگتر هست یا خیر. اگو بزرگتر بود، این به این معنی ست که این زیر رشته، بلندترین زیر رشته مشترک است.
در خط بعد با سرچ کردن در زبررشته ها ایندکس شروع آنها را پیدا میکنیم.
سپس چک میکنیم که زیررشته پیدا کرده ایم یا خیر. اگر نکردیم، ،،،۰ برمیگردانیم و اگر پیدا کردیم، طبق خواسته مساله یک توپل برمیگردانیم.

در سلول دوم در واقع در یک حلقه همیشه درست، ورودی را از کاربر میگیریم و longest_substring را به عنوان خروجی برمیگردانیم.

سوال انتخابی: سوال پنجم:

در سلول اول کتابخانه pandas را ایمپورت میکنیم.

در سلول دوم دیتاست را از حافظه میخوانیم و چک های لازم را انجام میدهیم که ببینیم data نیاز به تمیز کردن دارد یا خیر.

در سلول سوم نمایی کلی از دیتامان را میبینیم.

در سلول چهارم(پاسخ بخش اول سوال) با groupby میآییم و جمع همه ی دفعاتی که موزیک های مختلف شنیده شده اند را پیدا میکنیم. سپس به ترتیب ندولی آنها را سورت میکنیم که پربازدید ترین آهنگ ها را پیدا کنیم.

در سلول پنجم (پاسخ بخش دوم سوال) تعداد یوزر های منحصر به فردمان را پیدا میکنیم.

* برای پاسخ به بخش سوم سوال (سیستم پیشنهاد دهنده آهنگ) بنده پس از فکر کردن ساده ترین راهی که به ذهنم رسید این بود که خواننده مورد علاقه هر یوزر را بر اساس آهنگ هایی که بیشتر از همه به آنان گوش داده پیدا کنم. و سپس آهنگ هایی از آن خواننده که بیشترین دفعات شنیده شدن را دارند را به یوزر پیشنهاد دهم. در واقع باید برای این کار پرشنونده ترین آهنگ های هر آرتیست را پیدا و خواننده مورد علاقه هر یوزر را پیدا و سپس این دو را به هم وصل کنم.

در سلول هفتم، invalid row ها را پیدا کردم که شامل موزیک هاییست که در ستون song بیشتر از یک '-' دارند. شاید منطقی به نظر نرسد ولی در پروسه دیباگینگ مجبور به این کار شدم. چون تعدادشان نیز کم هست تاپیری در خروجی ندارد.

در سلول هشتم، فرایند تمیزکردن دیتاست را تکمیل کردم.

در سلول ۹ ام ستون title , artist را با توجه به دیتاست تمیز شده rewrite کردم. و ستون song را که دیگر نیازی نداریم حذف کردم. در نهایت head , tail دیتاست نهایی را چک کردم.

در سلول ۱۱ ام پرشنونده ترین(محبوب ترین) خواننده ها را پیدا کردم به همراه تعداد دفعاتی که موزیک هایشان شنیده شده. شامل جمع کردن تعداد دفعاتی که آهنگ هایشان شنیده شده و سپس سورت کردن دیتاست بر اساس ستون listen_count.

در سلول ۱۲ ام یک دیکشنری خالی ساختم که قرار است key های آن اسم خواننده ها و value هر کدام پنج آهنگ برتر آن خواننده با توجه به دفعات شنیده شدن باشد. سپس مقادیر این دیکشنری را پر کردم. و در سلول بعدی یکی از آرتیست ها را بصورت رندوم محبوب ترین آهنگ هایش را دیدم.

در سلول ۱۴ محبوب ترین خواننده هر یوزر را پیدا کردم بر اساس خواننده آهنگ هایی که گوش داده.

در سلول بعدی برای یوزر های مختلف متد idxmax() را اپلای میکنیم که یک سری با ماکسیمم مقدار برای هر ستون پیدا میکند.

در سلول بعدی از آبجکت idx که در سلول قبلی ساختم، لامبدا فانکشن اپلای کردم که آرتیست مورد علاقه یوزر را بیرون بکشد.

در سلول بعد، یک دیکشنری به نام user_data میسازیم که هر کلید آن یوزر آیدی هر کاربر و هر value آن آهنگ های مورد علاقه بیست که احتمالا دوست دارد. از idx سلول های قبلی برای یافتن تعداد دفعات شنیده شده هر ارتیست در دیتافریم مرتب شده استفاده میکنیم. از zip برای ترکیب کردن آرتیست های محبوب و دفعات شنیده شرنشان استفاده کردم تا لیستی از توپل ها بسازم. از list comprehension هم استفاده کردم که لیستی از دیکشنری ها بسازم که هر دیکشنری حاوی ارتیست مورد علاقه هر یوزر و listen count هست.

در سلول بعدی یک نمونه از داده ای که در سلول قبلی ساختم را چک کردم.

سپس در سلول بعدی یک تابع recommender ساختم که آیدی یوزر را میگیرد و آرتیست مورد علاقه اش را پیدا و در نهایت ترک های مورد علاقه اش را پرینت میکند.
در سلول های بعدی نیز چند نمونه از یوزر ها را چک کردم.