یک

الف) PCA بهتر است، چراکه برای n های پایین‌تر عملکرد آن از LDA بهتر است و درخت تک تصمیمی نیز برای m=50 نیز اصلا پاسخ‌گو نیست.

ب) چون تعداد داده‌های در هر دسته زیاد است و تمامی label ها نیز موجود است، الگوریتم LDA بهتر عمل می‌کند. هرچند چون این الگوریتم مشکلاتی با ویژگی‌های categorical دارد شاید در انتها نیز بتوان برای آن‌ها از درخت تصمیم استفاده کرد.

پ) برای داده‌های بدون برچسب مجبوریم از PCA استفاده کنیم، به علت بالا بودن n از LDA برای داده‌های برچسب‌دار استفاده می‌کنیم و سپس با تشکیل دادن Correlation matrix بین خروجی PCA و خروجی LDA، دسته‌بندی‌های مشابه را ادغام می‌کنیم.

دو. الگوریتم K-Means++ یک روش برای پیدا کردن Centroidهای اولیه برای الگوریتم K-Means است. در این روش ابتدا یک Centroid به صورت رندوم انتخاب می‌شود، سپس Centroid های بعدی جوری انتخاب می‌شوند که احتمال انتخاب آن‌ها، رابطه مستقیم با فاصله‌شان از نزدیک‌ترین Centroid داشته‌باشد. این‌کار را این‌قدر ادامه می‌دهیم تا همه‌ی Kتا Centroid پیدا شوند. این کار احتمال انتخاب Centroidهایی که متعلق به Clusterهای متفاوت هستند را مرحله به مرحله افزایش می‌دهد، پس در نهایت احتمالا Clusterهای بهتری خواهیم داشت.

سه.

الف) بله. طبق تعریف نقاط یک خوشه باید توسط Core Point آن خوشه، Reachable باشند و یک مسیر از آن Core Point به هرکدام آن‌ها وجود داشته‌باشد.

ب) خیر. بدترین حالت آن از اردر O(N^2) است. این پیچیدگی به تعداد صدا شدن تابع regionQuery بستگی دارد که با انتخاب اپسیلون معنی‌دار، این تابع در اردر log n نقطه برمی‌گرداند و در بدترین حالت در اردر n نقطه برمی‌گرداند که در بهترین حالت برای کل نقاط داریم O(N log N) و در بدترین حالت O(N^2).

پ) بله. چون نقاط پرت unreachable خواهند بود، در ساخت clusterهای ما مشکل خاصی ایجاد نمی‌کنند.

ت) درست است. برخلاف الگوریتم Spectral Clustering، این الگوریتم تنها بخش‌های Connected گراف reachability را پیدا می‌کند و نیازی به انتخاب k از قبل ندارد.

چهار.

الف) کم‌ترین فاصله بین خوشه‌های تکی اولیه برای A و B است (=۰.۱۲) پس آن‌ها را ترکیب می‌کنیم:

	AB	C	D	E	F
AB	0				
C	min[0.25, 0.51] = 0.25	0			
D	min[0.16, 0.84] = 0.16	0.14	0		
E	min[0.28, 0.77] = 0.28	0.7	0.45	0	
F	min[0.34, 0.71] = 0.34	0.93	0.2	0.67	0

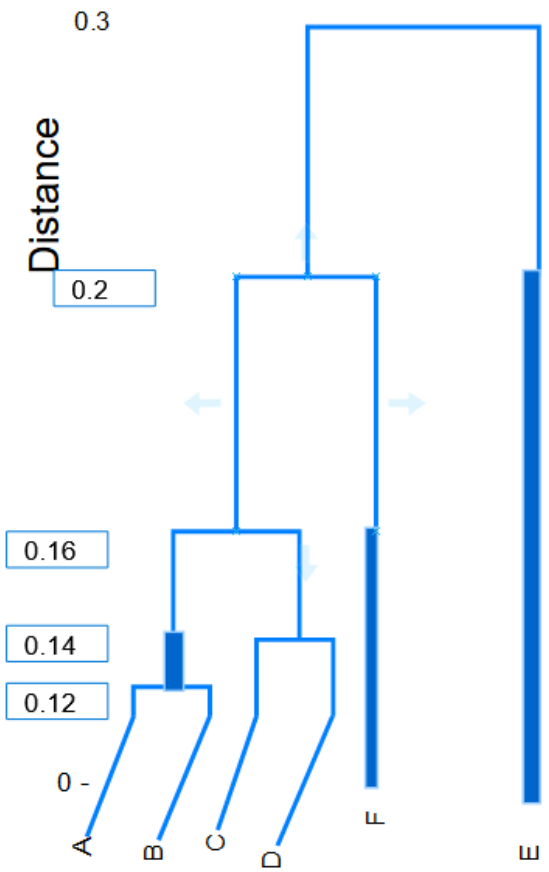
به همین ترتیب خوشه‌های با کم‌ترین فاصله را ترکیب می‌کنیم:

	AB	CD	E	F
AB	0			
CD	min[0.16, 0.25] = 0.16	0		
E	0.28	min[0.45, 0.7] = 0.45	0	
F	0.34	min[0.2, 0.93] = 0.2	0.67	0

	ABCD	E	F
ABCD	0		
E	min[0.28, 0.45] = 0.28	0	
F	min[0.2, 0.93] = 0.2	0.67	0

	ABCDF	E
ABCDF	0	
E	min[0.28, 0.67] = 0.28	0

نمودار dendogram آن نیز به این صورت در می‌آید:



ب) در این قسمت در هر مرحله خوشه‌های با فاصله‌ی مینیمم را ترکیب می‌کنیم ولی بر خلاف قسمت قبل، ماکسیمم فاصله‌ی بین خوشه‌ها را به عنوان فاصله‌ی جدید آن‌ها در نظر می‌گیریم و

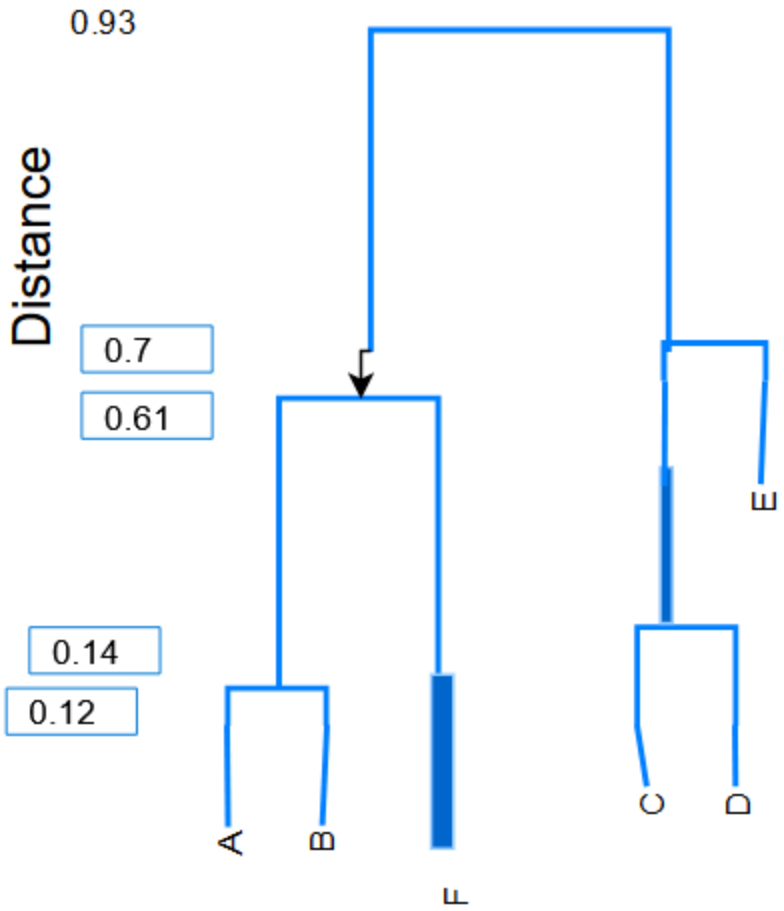
	AB	C	D	E	F
AB	0				
C	max[0.25, 0.51] = 0.51	0			
D	max[0.16, 0.84] = 0.84	0.14	0		
E	max[0.28, 0.77] = 0.77	0.7	0.45	0	
F	max[0.34, 0.71] = 0.61	0.93	0.2	0.67	0

	AB	CD	E	F
AB	0			
CD	max[0.51, 0.84] = 0.84	0		
E	0.77	max[0.45, 0.7] = 0.7	0	
F	0.61	max[0.2, 0.93] = 0.93	0.67	0

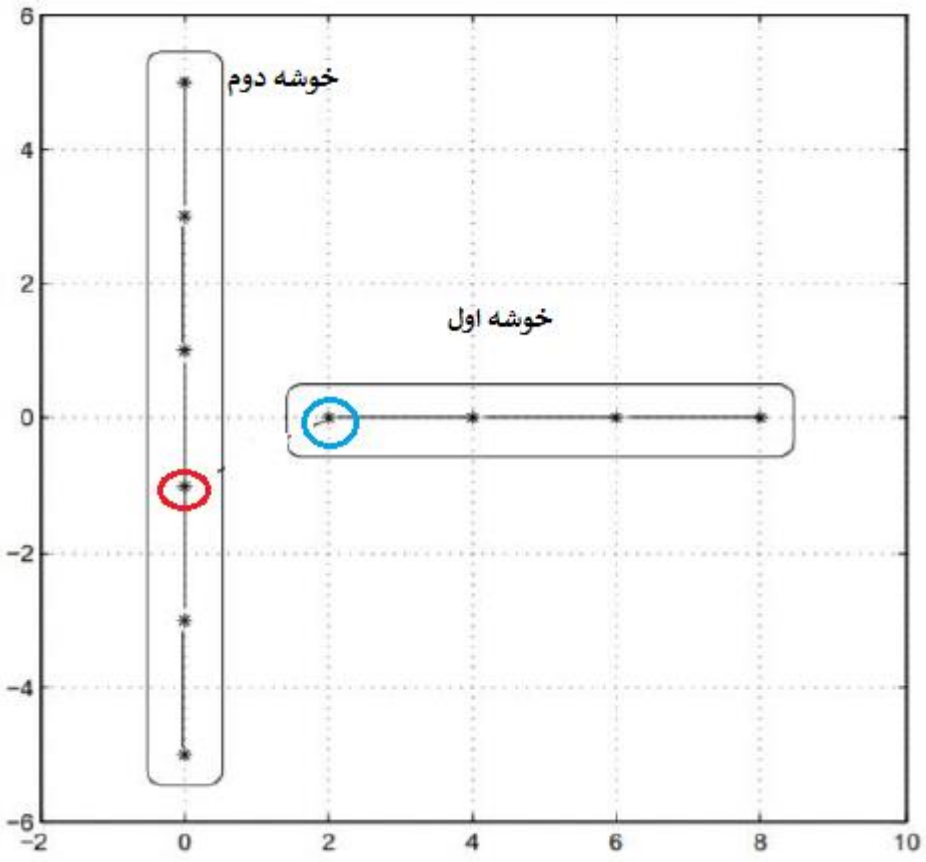
	ABF	CD	E
ABF	0		
CD	max[0.84, 0.93] = 0.93	0	
E	max[0.67, 0.77] = 0.77	0.7	0

	ABF	CDE
ABF	0	
CDE	max[0.77, 0.93] = 0.93	0

نمودار dendogram:


0.93
Distance
0.7
0.61
0.14
0.12
A
B
F
C
D
E

پنج.



الف)

ندی که با قرمز مشخص شده فاصله‌اش با ندهای بالا و پایین‌ترش هرکدام ۱ است و با ند سمت راستش، رادیکال دو، پس یالش به آن ند قطع می‌شود و این دو دسته را خواهیم داشت.

ب) در الگوریتم k-means ندی که با آبی مشخص شده نیز جزیی از خوشه‌ی دوم خواهد بود، چراکه فاصله‌اش با ندهای نزدیک‌ترش در خوشه‌ی دوم هر کدام رادیکال دو است اما فاصله‌اش با نزدیک‌ترین ند در خوشه‌ی اول ۲ است. پس الگوریتم k-means آن را جزیی از خوشه‌ی دوم حساب می‌کند.