سوال ۱ –

قسمت آ) اگر ماتریس داده هایمان، ماتریس X به سایز m*n باشد، که m تعداد نمونه ها و n تعداد ویژگی ها(ابعاد) باشد، ماتریس کوواریانس داده ها، یعنی ماتریس C به سایز n*n برابر است با:

$$C = \frac{X^T X}{n-1}$$

چون ماتریس کوواریانس یک ماتریس متقارن است، میتوان آن را به شکل قطری نوشت:

$$C = VLV^T$$

که V همان ماتریس متشکل از بردار های ویژه(هر ستون ماتریس V یک بردار ویژه است) و L یک ماتریس قطری متشکل از مقادیر ویژه به طور نزولی روی قطر های آن است.بردار های ویژه را "**محور های اصلی**" یا "**جهت های اصلی**" میگویند.تصویر داده هایمان روی محور های اصلی را هم مولفه های اصلی میگویند. برای مثال j امین مولفه اصلی داده ها همان j امین ستون ماتریس XV است.

ما در تجزیه مقدار تکین ماتریس X داریم:

$$X = USV^T$$

که U یک ماتریس یکانی و S یک ماتریس قطری متشکل از مقادیر تکین s_i هستند. لذا اگر بخواهیم ماتریس کوواریانس داده هایمان را بر اساس مقادیر تکین بنویسیم:

$$C = \frac{X^T X}{n-1} = \frac{VSU^T USV^T}{n-1} = V\frac{S^2}{n-1}V^T \rightarrow L = \frac{S^2}{n-1} \rightarrow \lambda_i = \frac{s_i^2}{n-1}$$

لذا مولفه های اصلی ماتریس داده مان میشود ستون های ماتریس زیر:

$$XV = USV^T V = US$$

قسمت ب) اگر تعداد نمونه ها N و ابعاد آن D باشد، وقتی در PCA از محاسبه ماتریس کوواریانس و سپس تجزیه مقادیر ویژه از آن استفاده کنیم، هزینه محاسباتی برابر است با :

۱- محاسبه ماتریس کوواریانس از مرتبه $O(ND * \min(N, D))$

۲- تجزیه مقادیر ویژه از آن از مرتبه $O(D^3)$

لذا مرتبه کلی این روش برابر است با:

$$O(ND * \min(N, D) + D^3)$$

حال اگر از SVD در PCA استفاده کنیم، مرتبه آن میشود:

$$O(\min\{N^2D, ND^2\})$$

لذا در شرایطی که D خیلی بزرگ تر از N باشد، مرتبه روش اول برابر است با $O(D^3)$ و مرتبه روش دوم برابر است با $O(N^2D)$ لذا قطعا استفاده از SVD بهینه تر است. از طرفی طبق کورس ماشین لرنینگ Andrew ng ، استفاده از SVD در PCA نسبت به استفاده از بردار ویژه های ماتریس کوواریانس، از لحاظ عددی بهینه تر و پایدار تر است و دقت بالاتری دارد و گمشدگی کمتری نیز دارد.

سوال ۲-

قسمت آ) برای اثبات این بخش، ابتدا به صورت شهودی اثبات را انجام میدهیم و سپس به صورت ریاضی هم آن را اثبات میکنیم. میدانیم که تعداد حالات مختلف خوشه بندی در الگوریتم k-means با n داده برابر است با k^n که عدد بزرگی میتواند باشد اما مهم این است که این عدد محدود است. طبق مفهوم الگوریتم k-means میدانیم زمانی این الگوریتم پایان میابد که هیچ نقطه ای به مرکز خوشه ای به جز خوشه ای که در آن زمان بهش تعلق داره، نزدیک تر نباشه. یعنی خوشه بندی به حالت پایدار رسیده باشه. لذا هیچ گاه امکان ندارد خوشه بندی مرحله بعدی برابر با خوشه بندی مرحله قبلی باشد زیرا در آن صورت در همان مرحله قبلی، الگوریتم باید پایان میافت. حال باید ثابت کنیم که با رفتن به مرحله بعدی(آپدیت خوشه ها و سپس آپدیت مرکز خوشه ها)، قطعا هزینه خوشه بندی کمتر از قبل میشود یعنی:

$$cost(C^{t+1}, \mu^{t+1}) < cost(C^t, \mu^t)$$

با اثبات این نامساوی، میتوان ادعا کرد همیشه هزینه نزولی خواهد بود و چون حداقل هزینه صفر است و تعداد حالات مختلف خوشه بندی هم محدود است، لذا قطعا این الگوریتم همگرا میشود.

برای اثبات این نامساوی، ابتدا نامساوی زیر را اثبات میکنیم:

$$cost(C^{t+1}, \mu^t) < cost(C^t, \mu^t)$$

و سپس نامساوی زیر را:

$$cost(C^{t+1}, \mu^{t+1}) < cost(C^{t+1}, \mu^t)$$

نامساوی اول که بدیهی است زیرا در صورتی الگوریتم پایان نمیابد و به مرحله بعد میرود که نقطه ای پیدا شود که به یک مرکز خوشه دیگری نزدیک باشد و خوشه ها باید آپدیت شود که در این صورت هزینه قطعا کمتر خوشه بندی مرحله قبل میشود یعنی:

$$cost(C^{t+1}, \mu^t) = \sum_{i=1}^{n} \left\| x^i - \mu^t_{C^{t+1}(i)} \right\|^2 < \sum_{i=1}^{n} \left\| x^i - \mu^t_{C^t(i)} \right\|^2 = cost(C^t, \mu^t)$$

برای اثبات نامساوی دوم، ابتدا لم زیر را تعریف میکنیم:

لم: اگر m نقطه به نام های $z_1, z_2, \dots, z_m$ داشته باشیم و میانگین آن ها را $\bar{z}$ بنامیم، با داشتن نقطه فرضی z در همان فضا، داریم:

$$\sum_{i=1}^{m} \|z_i - z\|^2 \geq \sum_{i=1}^{m} \|z_i - \bar{z}\|^2$$

لذا برای اثبات نامساوی دوم داریم:

$$cost(C^{t+1}, \mu^{t+1}) = \sum_{i=1}^{n} \left\| x^i - \mu^{t+1}_{C^{t+1}(i)} \right\|^2$$

$$= \sum_{k'=1}^{k} \sum_{i \in \{1,2,\dots,n\}, c^{t+1}(i)=k'} \left\| x^i - \mu^{t+1}_{C^{t+1}(i)} \right\|^2$$

$$\leq \sum_{k'=1}^{k} \sum_{i \in \{1,2,\dots,n\}, c^{t+1}(i)=k'} \left\| x^i - \mu^{t}_{C^{t+1}(i)} \right\|^2$$

$$= \sum_{i=1}^{n} \left\| x^i - \mu^{t}_{C^{t+1}(i)} \right\|^2 = cost(C^{t+1}, \mu^t)$$

قسمت ب) با فرض اینکه مقداردهی اولیه مراکز در فضا به شکل یکنواخت باشد، یعنی حالت های خاص را در نظر نگیریم، بخاطر اینکه نیمی از داده ها در ناحیه ای به شکل dense وجود دارند، در اغلب موارد، نزدیک ترین مرکز(در مقداردهی اولیه) آن داده هارا یک خوشه میکند و بخاطر نزدیکی فاصله بین داده های آن ناحیه، مراکز دیگر توانایی شکستن آن خوشه را ندارند

و مشغول ساختن خوشه در ناحیه sparse میشوند. این که گفتم یک مرکز یک خوشه از ناحیه desne میسازد، در حالت عمومی میتواند بیشتر از یک مرکز و بیشتر از یک خوشه باشد(بستگی به میزان dense بودن و مقداردهی اولیه مراکز دارد) اما در حالت کلی منظورم این بود که چگالی مراکز در ناحیه dense در صورت وقوع شرط اولیه ، قطعا کمتر از چگالی مراکز در ناحیه sparse است زیرا در ناحیه sparse داده ها با هم فاصله دارند و مراکز مختلف نزدیک ترین داده هارا مال خود میکنند و خوشه تشکیل میدهند و در آپدیت های الگوریتم، ممکن است مراکز جدید به ناحیه sparse بروند و خوشه های بیشتری تشکیل دهند اما ناحیه dense بخاطر فاصله کم داده ها، در تعداد کمی آپدیت، متعلق به یک یا چند مرکز(تعداد کم) میشوند و مراکز دیگر نمیتوانند خوشه متمرکز را بهم بزنند زیرا مجموع مربع خطا در آن خوشه بسیار کم است و نقاط به مرکز گفته شده بسیار نزدیک اند و بهترین حالت برای آن ها رخ داده است.در یک کلام، مراکز در کل فضا به صورت تقریبا یکنواخت پخش میشوند و بخاطر مساحت بیشتر ناحیه غیرمتراکم، چگالی مراکز در آن ناحیه بیشتر از ناحیه متراکم است.

قسمت پ) الگوریتم ذکر شده در سوال همان الگوریتم k-means++ است که فقط در مقداردهی اولیه مرکز ها با k-means عادی متفاوت است. در k-means عادی، حساسیت خیلی زیادی به مقداردهی اولیه مراکز وجود دارد و ممکن است یک خوشه بندی بسیار بد بوجود بیاورند(مثلا وقتی که ۳ ناحیه متمرکز در صفحه به شکل مثلثی داشته باشیم و مقدار دهی اولیه مراکز به این صورت باشد که یک مرکز در مرکز این مثلث قرار بگیرد و دو مرکز دیگر، در گوشه های صفحه باشند، در این حالت هر سه ناحیه به مرکز اول انتصاب پیدا میکنند و دو مرکز دیگر، حتی احتمالا بدون هیچ نقطه ای میشود و الگوریتم با یک خوشه که تمام داده هارا دارد و دو خوشه بدون داده پایان میابد).

حال در الگوریتم گفته شده، ابتدا مرکز اول به صورت رندوم در صفحه انتخاب میشود و در انتخاب مراکز بعدی، دو چیز را رعایت میکنیم: ۱- اینکه مراکز بعدی از بین نقاط انتخاب شود، بخاطر اینکه هیچ مرکزی بدون نقطه نشود ۲-مراکز تا جایی که امکان دارند از هم دور باشند، اینگونه خوشه های بهتری نسبت به الگوریتم عادی نتیجه میشود

این الگوریتم نسبت به الگوریتم عادی سریع تر همگرا میشود، دقت بهتری دارد، پیچیدگی محاسباتی کمتری دارد(بخاطر همگرایی سریع تر) و اینکه نتیجه نهایی میتواند(اغلب) با نتیجه نهایی الگوریتم عادی متفاوت باشد. توجه شود که بخاطر انتخاب رندوم مرکز اول هم در الگوریتم عادی هم در الگوریتم گفته شده در صورت سوال، همیشه میتوانیم در هر حالتی حتی در مقایسه با rerun های مختلف یک الگوریتم هم نتایج مختلفی بدست آوریم چه برسد به نتیجه الگوریتم k-means و k-means++ .

سوال ۳ –

قسمت آ) ابتدا Q-table را تشکیل میدهیم:

State-action	up	right	down	left
S11	-	0.5	0.5	-
S12	-	0.5	0.5	0.5
S13	-	-	0	0
S21	0.5	0.5	0.5	-
S22	0.5	0.5	0.5	0.5
S23	0.5	-	0.5	0.5
S31	0	0	-	-
S32	0.5	0.5	-	0.5
S33	0	-	-	0

حال دنباله اول را مرحله به مرحله بررسی میکنیم: $s_{11} \rightarrow s_{12} \rightarrow s_{13}$

ابتدا از *s11* به *s12* میرویم یعنی عمل *right* را انجام میدهیم.

حال داریم:

$$new\ Q(s_{11}, right) = old\ Q(s_{11}, right)$$

$$+\alpha[R(s_{11}, right) + \gamma(\max\{Q'(s_{12}, a')\} - Q(s_{11}, right\}]$$

$$\rightarrow new\ Q(s_{11}, right) = 0.5$$

$$+\alpha[0 + \gamma(\max\{Q(s_{12}, right), Q(s_{12}, down), Q(s_{12}, left)\}) - 0.5]$$

$$= 0.5 + 0.5\alpha\gamma - 0.5\alpha = 0.5(1 + \alpha\gamma - \alpha)$$

لذا داریم:

State-action	up	right	down	left
S11	-	$0.5(1 + \alpha\gamma - \alpha)$	0.5	-
S12	-	0.5	0.5	0.5
S13	-	-	0	0
S21	0.5	0.5	0.5	-
S22	0.5	0.5	0.5	0.5
S23	0.5	-	0.5	0.5
S31	0	0	-	-
S32	0.5	0.5	-	0.5
S33	0	-	-	0

حال در قدم بعدی داریم:

$$new\ Q(s_{12}, right) = old\ Q(s_{12}, right)$$
$$+\alpha[R(s_{12}, right) + \gamma(\max\{Q'(s_{13}, a'\}) - Q(s_{12}, right)]$$
$$\rightarrow new\ Q(s_{12}, right) = 0.5$$
$$+\alpha[-1 + \gamma(\max\{Q(s_{13}, down), Q(s_{13}, left)\}) - 0.5]$$
$$= 0.5 - \alpha - 0.5\alpha = 0.5(1 - \alpha) - \alpha$$

لذا داریم:

State-action	up	right	down	left
S11	-	$0.5(1 + \alpha\gamma - \alpha)$	0.5	-
S12	-	$0.5(1 - \alpha) - \alpha$	0.5	0.5
S13	-	-	0	0
S21	0.5	0.5	0.5	-
S22	0.5	0.5	0.5	0.5
S23	0.5	-	0.5	0.5
S31	0	0	-	-
S32	0.5	0.5	-	0.5
S33	0	-	-	0

حال دنباله بعدی را تحلیل میکنیم: $s_{11} \rightarrow s_{12} \rightarrow s_{22} \rightarrow s_{32} \rightarrow s_{33}$

ابتدا داریم:

$$new\ Q(s_{11}, right) = old\ Q(s_{11}, right)$$
$$+\alpha[R(s_{11}, right) + \gamma(\max\{Q'(s_{12}, a'\}) - Q(s_{11}, right\}]$$
$$\rightarrow new\ Q(s_{11}, right) = 0.5$$
$$+\alpha[0 + \gamma(\max\{Q(s_{12}, right), Q(s_{12}, down), Q(s_{12}, left)\}) - 0.5]$$
$$= 0.5(1 - \alpha)^2 + 0.5\alpha\gamma(2 - \alpha)$$

لذا داریم:

State-action	up	right	down	left
S11	-	$0.5(1-\alpha)^2 + 0.5\alpha\gamma(2-\alpha)$	0.5	-
S12	-	$0.5(1-\alpha)-\alpha$	0.5	0.5
S13	-	-	0	0
S21	0.5	0.5	0.5	-
S22	0.5	0.5	0.5	0.5
S23	0.5	-	0.5	0.5
S31	0	0	-	-
S32	0.5	0.5	-	0.5
S33	0	-	-	0

حال در قدم بعدی داریم:

$$new\ Q(s_{12}, down) = old\ Q(s_{12}, down)$$

$$+\alpha[R(s_{12}, down) + \gamma(\max\{Q'(s_{22}, a'\}) - Q(s_{12}, down)]$$

$$\rightarrow new\ Q(s_{12}, down) = 0.5$$

$$+\alpha[0 + \gamma(\max\{Q(s_{22}, right), Q(s_{22}, down), Q(s_{22}, left), Q(s_{22}, up)\}) - 0.5]$$

$$= 0.5(1-\alpha) + 0.5\alpha\gamma$$

لذا داریم:

State-action	up	right	down	left
S11	-	$0.5(1-\alpha)^2 + 0.5\alpha\gamma(2-\alpha)$	0.5	-
S12	-	$0.5(1-\alpha)-\alpha$	$0.5(1-\alpha) + 0.5\alpha\gamma$	0.5
S13	-	-	0	0
S21	0.5	0.5	0.5	-
S22	0.5	0.5	0.5	0.5
S23	0.5	-	0.5	0.5
S31	0	0	-	-
S32	0.5	0.5	-	0.5
S33	0	-	-	0

سپس داریم:

$$new\ Q(s_{22}, down) = old\ Q(s_{22}, down)$$
$$+\alpha[R(s_{22}, down) + \gamma(\max\{Q'(s_{32}, a'\}) - Q(s_{22}, down)]$$
$$\rightarrow new\ Q(s_{22}, down) = 0.5$$
$$+\alpha[0 + \gamma(\max\{Q(s_{32}, right), Q(s_{32}, left), Q(s_{32}, up)\}) - 0.5]$$
$$= 0.5(1-\alpha) + 0.5\alpha\gamma$$

لذا داریم:

State-action	up	right	down	left
S11	-	$0.5(1-\alpha)^2 + 0.5\alpha\gamma(2-\alpha)$	0.5	-
S12	-	$0.5(1-\alpha) - \alpha$	$0.5(1-\alpha) + 0.5\alpha\gamma$	0.5
S13	-	-	0	0
S21	0.5	0.5	0.5	-
S22	0.5	0.5	$0.5(1-\alpha) + 0.5\alpha\gamma$	0.5
S23	0.5	-	0.5	0.5
S31	0	0	-	-
S32	0.5	0.5	-	0.5
S33	0	-	-	0

و در آخر داریم:

$$new\ Q(s_{32}, right) = old\ Q(s_{32}, right)$$
$$+\alpha[R(s_{32}, right) + \gamma(\max\{Q'(s_{33}, a'\} - Q(s_{32}, right\})]$$
$$\rightarrow new\ Q(s_{32}, right) = 0.5$$
$$+\alpha[1 + \gamma(\max\{Q(s_{33}, left), Q(s_{33}, up)\} - 0.5)]$$
$$= 0.5(1+\alpha)$$

پس داریم:

State-action	up	right	down	left
S11	-	$0.5(1-\alpha)^2 + 0.5\alpha\gamma(2-\alpha)$	0.5	-
S12	-	$0.5(1-\alpha)-\alpha$	$0.5(1-\alpha) + 0.5\alpha\gamma$	0.5
S13	-	-	0	0
S21	0.5	0.5	0.5	-
S22	0.5	0.5	$0.5(1-\alpha) + 0.5\alpha\gamma$	0.5
S23	0.5	-	0.5	0.5
S31	0	0	-	-
S32	0.5	$0.5(1+\alpha)$	-	0.5
S33	0	-	-	0

قسمت ب) همانطور که میدانیم در حالت کلی و بدون در نظر گرفتن حالت خاص، هرچه ضریب تخفیف به ۱ نزدیک باشد، عامل سود زودهنگام را کمتر میخواهد و به سود بلند مدت بیشتر فکر میکند و برعکس. لذا با توجه به اینکه با انتخاب ضریب تخفیف شماره ۱ ، سود بیشتری نصیب امان شده و با توجه به دنیای مسئله که سود بیشتر در گروی مسیر بیشتر(تا رسیدن به حالت *s33*) است، لذا میفهمیم ضریب تخفیف شماره ۱ بزرگ تر از شماره ۲ است و اینگونه عامل در حالت اول ، سود بلند مدت برایش مهم تر بوده و بیشتر سعی میکند به حالت *s33* برسد.

اما اگر بازی *deterministic* باشد، میتوان بهتر نسبت این دورا تحلیل کرد.وقتی قدم اول را برداریم و به خانه *s12* برویم، دو راه پیش رو هست که راه بهینه تر آن است که ۳ قدم دیگر برداریم و خود را به خانه*s33* برسانیم.لذا در مقایسه دو رویکرد داریم:

$$Q_1(s_{11}, right) = \gamma_1^3 \quad , \quad Q_2(s_{11}, right) = \gamma_2^3$$

$$\rightarrow \frac{\gamma_1}{\gamma_2} = \left(\frac{9}{7}\right)^3 \approx 2.13$$

قسمت پ) در روش های model-based مشکلی که اغلب در دنیای واقعی وجود دارد، این است که تابع پاداش و تابع انتقال ناشناخته است و باید تمام پاداش ها و انتقال های مختلف را به عنوان پارامتر یاد بگیریم. لذا به ازای هر حالت غیر پایانی به جز s11 و s12 ما ۱۸ پارامتر داریم.به ازای s11 ما ۸ پارامتر و به ازای s22 ما ۳۲ پارامتر داریم(هر زوج حالت و اکشن یک پاداش و یک احتمال دارد). بنابراین در کل ۱۱۲ پارامتر داریم.

در روش های model-free ، باید تمام Q-Value های ممکن را یاد بگیریم و با توجه به ساختار مسئله، داریم:

$$number\ of\ Q-values\ for\ s_{11}=2,$$
$$number\ of\ Q-values\ for\ S_{12}=3,$$
$$number\ of\ Q-values\ for\ s_{21}=3,$$
$$number\ of\ Q-values\ for\ s_{22}=4,$$
$$number\ of\ Q-values\ for\ s_{23}=3,$$
$$number\ of\ Q-values\ for\ s_{32}=3$$
$$\rightarrow \#\ of\ parameters\ to\ learn\ is\ 2+3+3+4+3+3=18$$

بنابراین روش *model-free* مثل *Q-learning* بهتر عمل میکند.