Մեքենայական ուսուցումից մինչև PCA

Դասի ճանապարհային քարտեզ

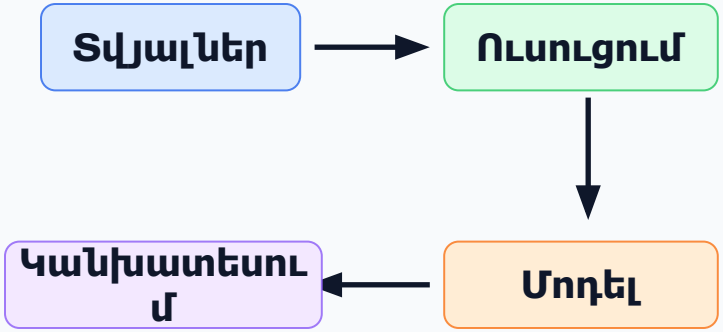
Սկսում ենք ընդհանուր ՄՈՒ-ից, հետո խորանում PCA-ի քայլերի մեջ

1. Ի՞նչ է մեքենայական ուսուցումը
2. Վերահսկվող / չվերահսկվող / ամրապնդմամբ ուսուցում
3. Չափողականության կրճատման խնդիրն ու կիրառությունները
4. PCA-ի մաթեմատիկան՝ միջին, դիսպերսիա, կովարիացիա, սեփական վեկտոր
5. Ձեռքով օրինակ և Python իրականացում

Ի՞նչ է մեքենայական ուսուցումը

ՄՈՒ-ն սովորեցնում է համակարգին օրինաչափություն գտնել տվյալներից

- Սովորական ծրագրում մարդը գրում է կանոնները:
- ՄՈՒ-ում մենք տալիս ենք օրինակներ, իսկ մոդելը սովորում է օրինաչափություն:
- Մոդելը հետո օգտագործվում է նոր, չտեսած տվյալների վրա:



օրինակներ → օրինաչափություն → որոշում

Տվյալների լեզուն. նմուշ, հատկանիշ, թիրախ

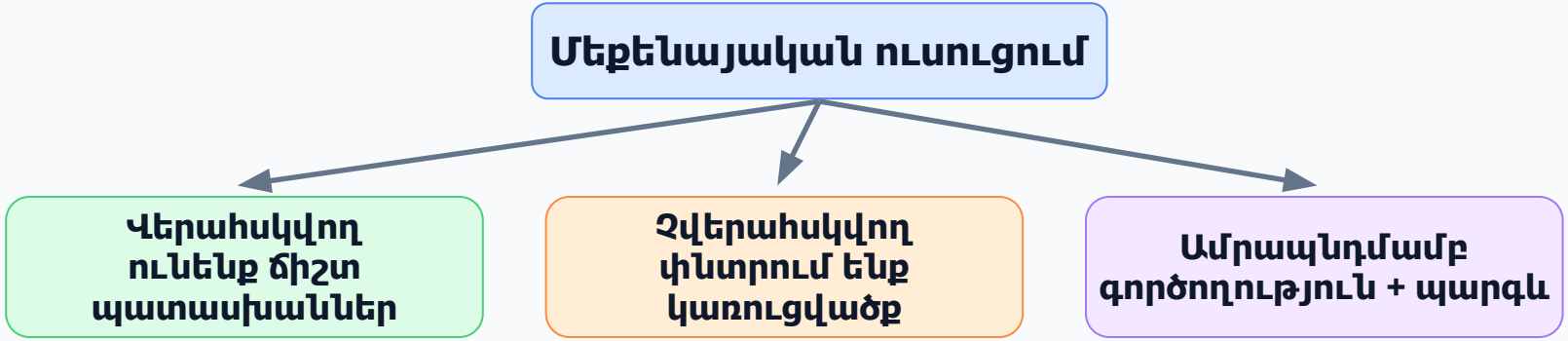
Նմուշ	հատկանիշ 1	հատկանիշ 2	թիրախ
տուն A	80 մ²	կենտրոն	75 մլն
տուն B	45 մ²	արվարձան	32 մլն
տուն C	120 մ²	կենտրոն	?

- Նմուշ՝ մեկ օբյեկտ կամ դեպք
- Հատկանիշ՝ թիվ/տեքստ, որը նկարագրում է նմուշը
- Թիրախ՝ պատասխանը, եթե այն ունենք

$X \in \mathbb{R}^{n \times d}$, y՝ թիրախների վեկտոր

ՄՈՒ-ի հիմնական ուղղությունները

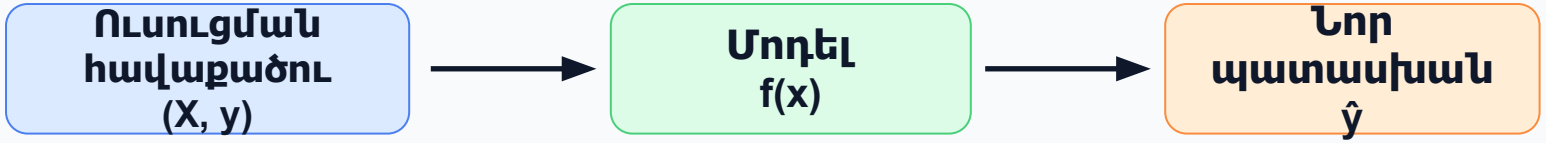
Խնդրի ձևը որոշում է ուսուցման տեսակը



Կիսավերահսկվող և ինքնավերահսկվող մեթոդները հաճախ միջանկյալ կամ հատուկ դեպքեր են

Վերահսկվող ուսուցում

Տվյալներ + ճիշտ պատասխան → սովորում ենք կանխատեսել



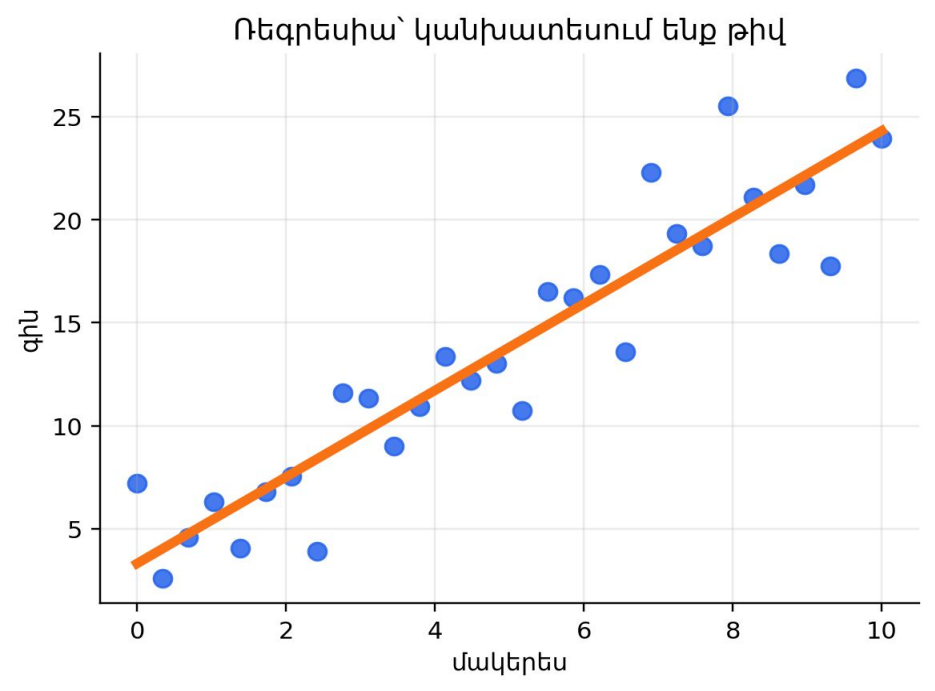
- Օրինակներ՝ գնի կանխատեսում, հիվանդության ռիսկ, սպամի հայտնաբերում
- Սխալը չափում ենք՝ ŷ-ը համեմատելով y-ի հետ
- Նպատակը՝ լավ աշխատել ոչ միայն հիշած, այլև նոր տվյալների վրա

Վերահսկվող ռեգրեսիա

Երբ պատասխանը թիվ է

- Թիրախը շարունակական մեծություն է՝ գին, ջերմաստիճան, միավոր:
- Մոդելը սովորում է կապը հատկանիշների և թվի միջև:
- Պարզ օրինակ՝ ուղիղ գիծ տվյալների միջով:

$$y \approx kx + b$$

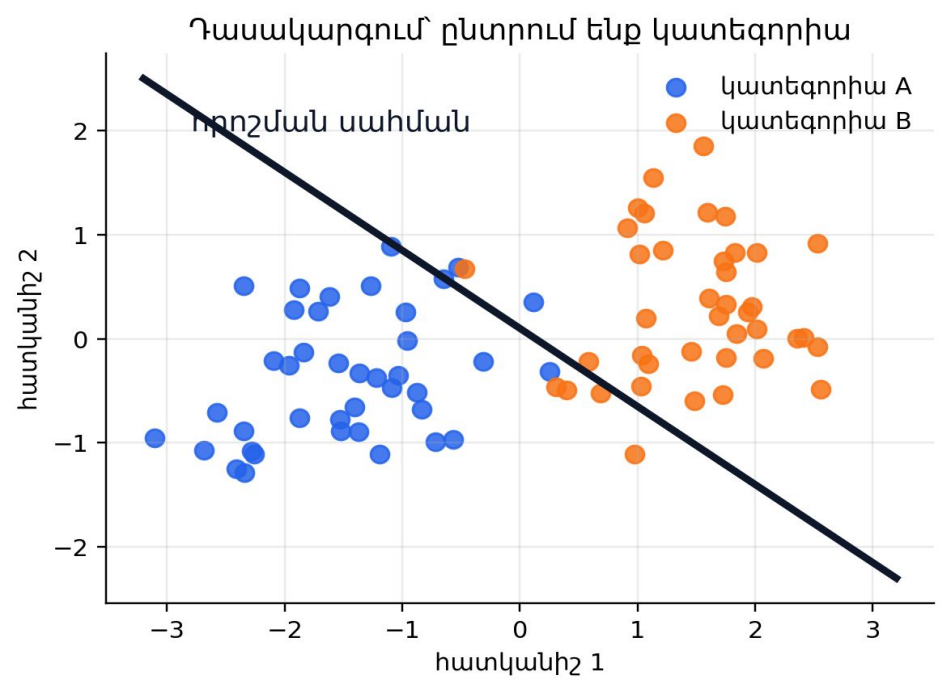


Վերահսկվող դասակարգում

Երբ պատասխանը կատեգորիա է

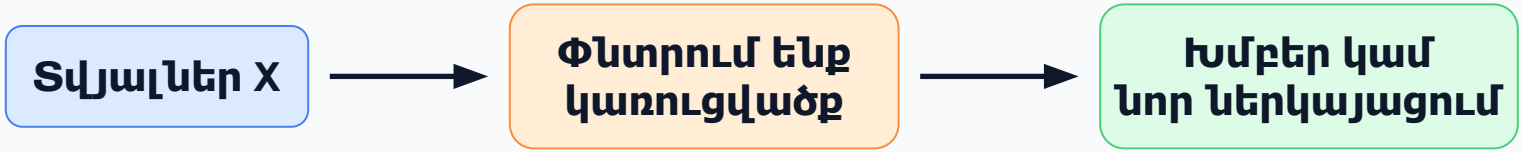
- Թիրախը դաս է՝ սպամ/ոչ սպամ, կատու/շուն, անցավ/չանցավ:
- Մոդելը սովորում է որոշման սահման:
- Սխալը՝ սխալ դասի ընտրությունն է:

$\hat{y} \in \{\text{դաս 1, դաս 2, …}\}$



Չվերահսկվող ուսուցում

Տրված են միայն X տվյալները. պատրաստի պատասխաններ չկան

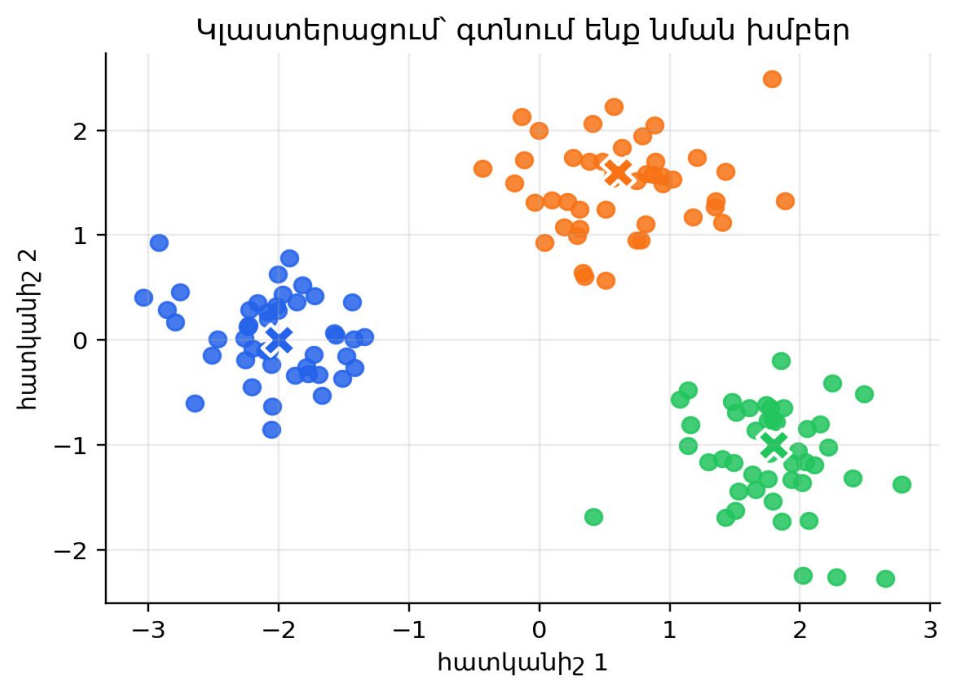


- Կլաստերացում՝ գտնել նման օբյեկտների խմբեր:
- Չափողականության կրճատում՝ շատ հատկանիշներ → քիչ կոորդինատներ:
- PCA-ն չվերահսկվող գծային չափողականության կրճատում է:

Չվերահսկվող կլաստերացում

Նման կետերը հավաքվում են նույն խմբում

- Օրինակ՝ հաճախորդների հատվածավորում:
- Խմբերը նախապես չեն տրված:
- Արդյունքը պետք է մեկնաբանել խնդրի համատեքստում:



Չափողականության կրճատումը ՄՈՒ քարտեզում

Չվերահսկվող ուսուցում → Չափողականության կրճատում → PCA

$$X \in \mathbb{R}^{n \times d} \rightarrow Z \in \mathbb{R}^{n \times k}, \quad k < d$$

- նմուշների թիվը՝ n, սովորաբար չի փոխվում
- փոխվում է հատկանիշների/կոորդինատների թիվը՝ d-ից k

Ի՞նչ է չափողականությունը

Չափողականությունը հատկանիշների թիվն է

d սյուն = հատկանիշներ

n տող = նմուշներ

28 × 28 պատկեր

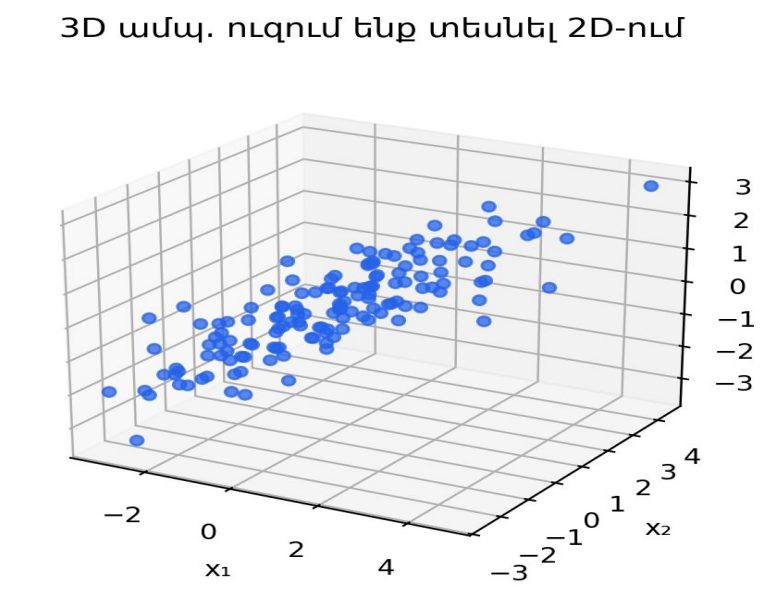
784 պիքսելային հատկանիշ

d = 784

- Եթե նկարն ունի շատ պիքսել, այն մեծ չափանի վեկտոր է:
- PCA-ն փորձում է այդ երկար վեկտորը ներկայացնել քիչ թվերով:

Ինչո՞ւ կրճատել չափողականությունը

Բարձր չափերում դժվար է տեսնել, պահել և սովորել



- Տեսապատկերում՝ 100 կամ 784 հատկանիշը չենք կարող ուղիղ նկարել:
- Սեղմում՝ նույն տեղեկությունը պահել ավելի քիչ թվերով:
- Աղմուկի նվազեցում՝ երբ փոքր դիսպերսիայի ուղղությունները հիմնականում աղմուկ են:
- Արագացում՝ հաջորդ մոդելը կարող է աշխատել ավելի քիչ հատկանիշով:

Բայց չափազանց մեծ կրճատումը կարող է կորցնել օգտակար ազդանշանը

Կիրառություններ

Չափողականության կրճատումը օգնում է տարբեր նպատակների համար

Տեսապատկերում
բարձր չափանի տվյալը նկարել 2D/3D-ում

Սեղմում
պահել քիչ թվեր, մոտավոր վերականգնել

Գերհարմարեցման նվազեցում
քիչ հատկանիշով հետագա մոդելը պարզ է

Բացակայող արժեքներ
ցածր ռանգով վերականգնումը կարող է օգնել լրացման մեջ

Կցված նյութի PCA սլայդում նշվում են visualization, feature selection և data imputation կիրառությունները:

Հատկանիշների ընտրությո՞ւն, թե՞ արտածում

PCA-ն սովորաբար ստեղծում է նոր հատկանիշներ

Ընտրություն
պահում ենք սկզբնական սյունակների մի մասը

→

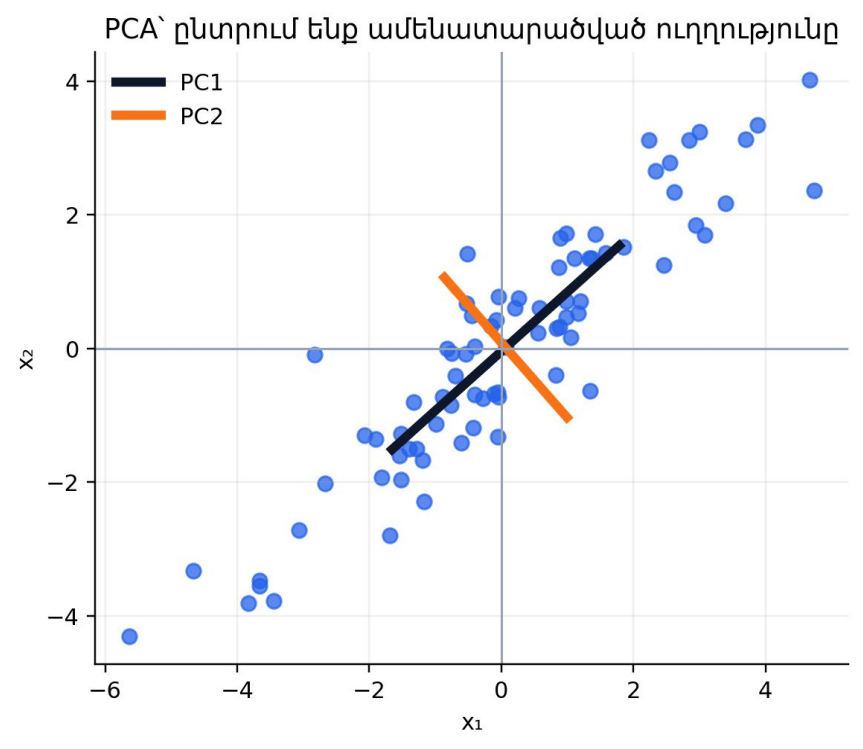
Արտածում
ստեղծում ենք նոր համակցություններ

PCA = feature extraction

$$z = 0.7 \cdot x_1 + 0.7 \cdot x_2$$

PCA-ի ինտուիցիան. գտնել երկար առանցքը

Լավ առանցքը պահում է կետերի տարածվածությունը



- Կետերը ձգված են մեկ թեք ուղղությամբ:
- Եթե պահենք այդ ուղղության կոորդինատը, քիչ տեղեկություն կկորցնենք:
- PCA-ն ընտրում է մեծագույն դիսպերսիայի ուղղությունը:

առաջին գլխավոր բաղադրիչ = PC1

Մաթեմատիկական գաղափար 1. միջին և դիսպերսիա

Միջինը ցույց է տալիս կենտրոնը, դիսպերսիան՝ տարածվածությունը

$$\mu = (x_1 + x_2 + \ldots + x_n) / n$$

միջին = կենտրոն

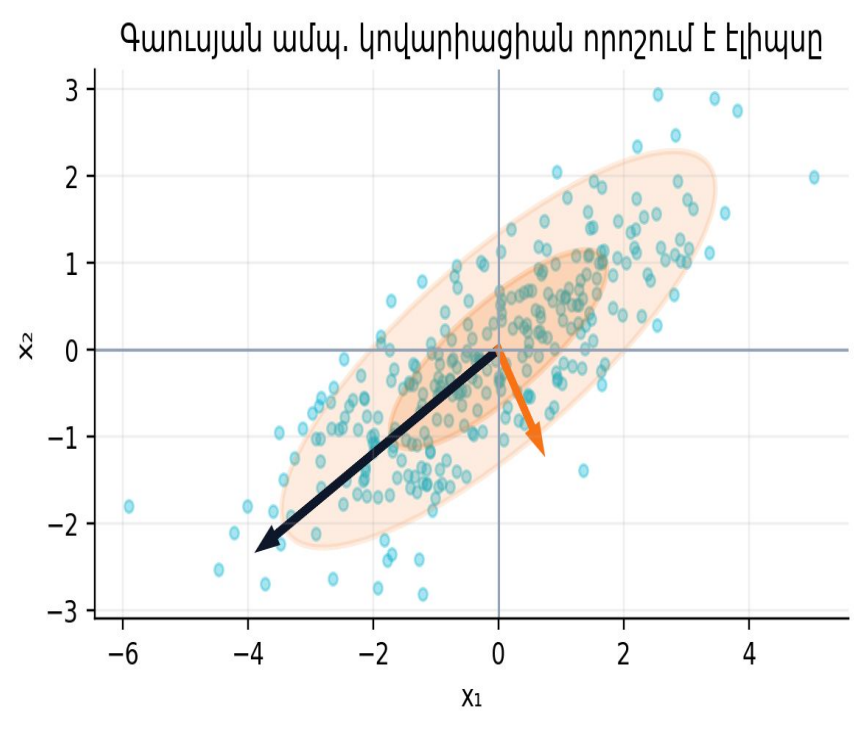
$$Var(x) = \Sigma(x_i - \mu)^2 / (n - 1)$$

դիսպերսիա = որքան են կետերը ցրված կենտրոնից

- PCA-ն սիրում է այն ուղղությունները, որտեղ դիսպերսիան մեծ է:
- Մեծ դիսպերսիա ≠ միշտ կարևոր է թիրախի համար, բայց PCA-ի չափանիշը դա է:

Մաթեմատիկական գաղափար 2. գաուսյան բաշխում

Զանգաձև ամպ, որի ձևը նկարագրվում է միջինով և կովարիացիայով



- 1D-ում գաուսյանը զանգի նման բաշխում է:
- 2D-ում այն հաճախ երևում է որպես էլիպսաձև կետերի ամպ:
- Էլիպսի երկար առանցքը ցույց է տալիս մեծ դիսպերսիայի ուղղությունը:
- PCA-ն պարտադիր չի պահանջում գաուսյանություն, բայց այս պատկերը օգնում է հասկանալ երկրաչափությունը:

$$N(\mu, C)$$

Մաթեմատիկական գաղափար 3. կովարիացիա

Կովարիացիան չափում է՝ երկու հատկանիշները միասի՞ն են փոխվում

$$\mathrm{Cov}(x,y) = \Sigma(x_i - \mu_x)(y_i - \mu_Y) / (n - 1)$$

դրական
մեծ x-ի հետ հաճախ մեծ y

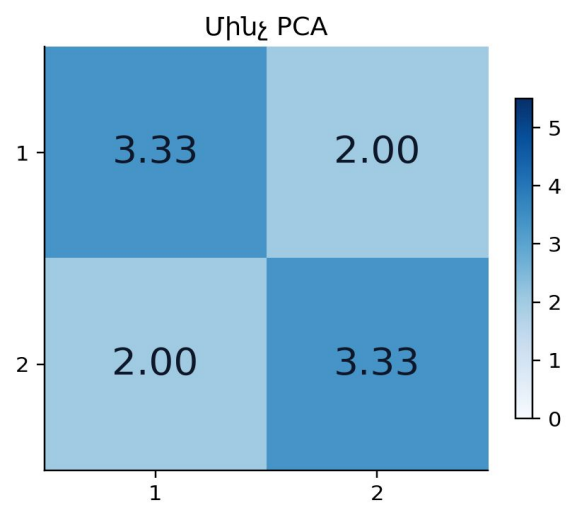
բացասական
մեծ x-ի հետ հաճախ փոքր y

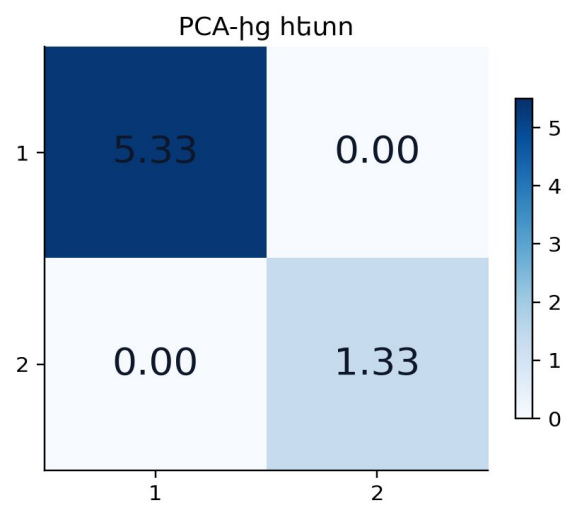
մոտ զրո
գծային կապը թույլ է

- Կովարիացիան կախված է չափման միավորներից:
- Կոռելյացիան կովարիացիայի սանդղակված տարբերակն է՝ −1-ից 1:

Մաթեմատիկական գաղափար 4. կովարիացիայի մատրից

Բոլոր զույգ կապերը դրվում են մեկ աղյուսակի մեջ





անկյունագիծ = դիսպերսիաներ

անկյունագծից դուրս = կովարիացիաներ

PCA քայլ 1. կենտրոնացնել և երբեմն ստանդարտացնել

PCA-ն ուսումնասիրում է շեղումները միջինից

$$Xc_{i□} = X_{i□} - μ_{□}$$

կենտրոնացում
յուրաքանչյուր սյան միջինը դարձնում է 0

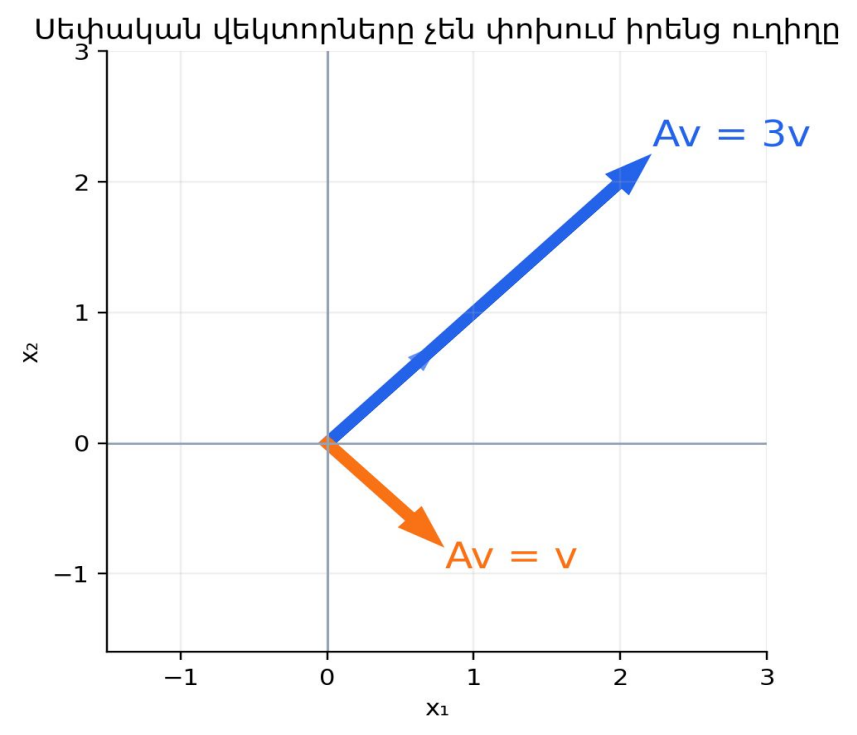
ստանդարտացում՝ $(X_{i□} - μ_{□}) / s_{□}$

ստանդարտացում
օգնում է, երբ միավորները տարբեր են

- Նոր տվյալների վրա օգտագործում ենք նույն μ-ն և s-ը, որոնք սովորել ենք train-ից:
- Օրինակ՝ դրամը և տարիքը նույն սանդղակի վրա չեն:

Մաթեմատիկական գաղափար 6. սեփական վեկտոր և արժեք

Սեփական վեկտորը ձևափոխումից հետո մնում է նույն ուղղի վրա



$$A\,v = \lambda\,v, \quad v \neq 0$$

- v՝ սեփական վեկտոր՝ հատուկ ուղղություն:
- λ՝ սեփական արժեք՝ որքան է ձգվում կամ սեղմվում:
- PCA-ում C-ի սեփական վեկտորները դառնում են գլխավոր առանցքներ:
- Մեծ λ նշանակում է մեծ դիսպերսիա այդ ուղղությամբ:

Տեսնենք սեփական վեկտորների շարժումը

3Blue1Brown տեսանյութ՝ ինտուիցիայի համար

Դիտելիս գտեք՝ որ վեկտորներն են մնում իրենց ուղղի վրա

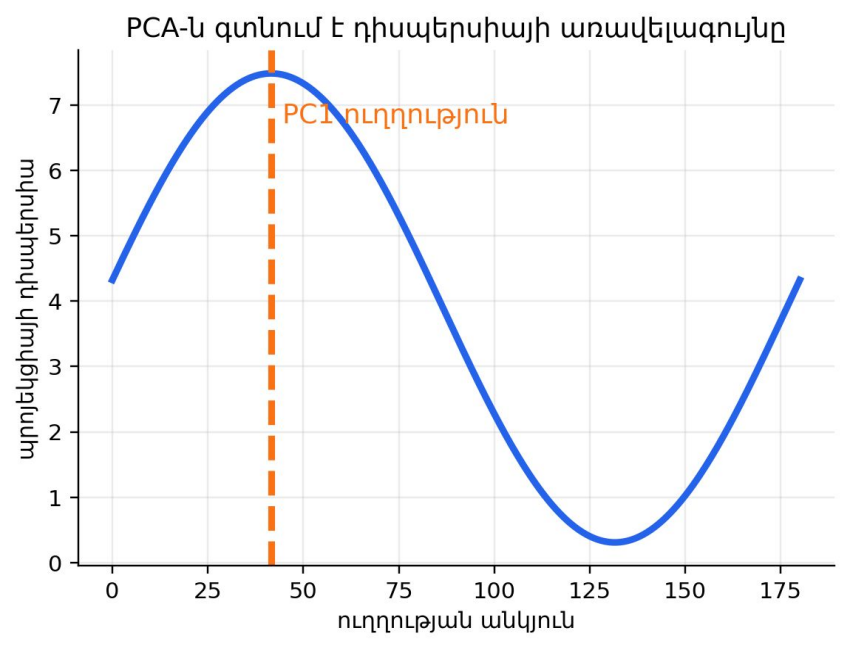
https://www.youtube.com/watch?v=PFDu9oVAE-g

Առաջարկ՝ դիտել մոտ 4 րոպե այն հատվածից, որտեղ ցանցը ձևափոխվում է և երևում են սեփական ուղղությունները:

Հարց քննարկման համար. λ-ն ի՞նչ է ասում երկարության մասին

PCA-ի նպատակը բանաձևով

Ընտրել ուղղություն, որտեղ պրոյեկցիաների դիսպերսիան մեծագույնն է



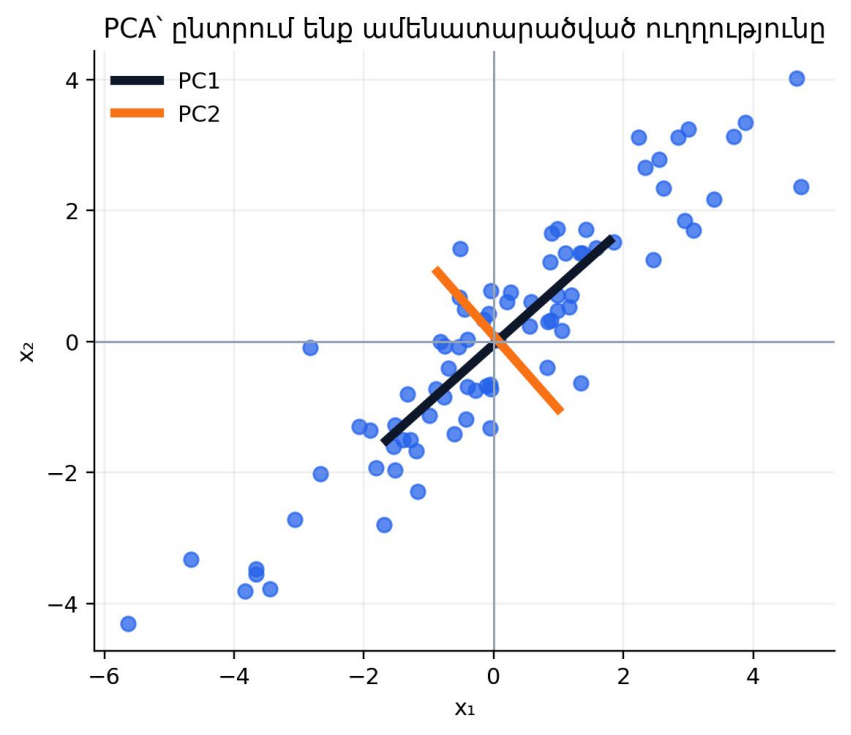
$$\text{maximize} \quad \mathrm{Var}(X_c\, w) = w^T C\, w$$

պայման՝ $\|w\| = 1$

- Առանց $\|w\|=1$ պայմանի կարող ենք w-ը մեծացնել և դիսպերսիան անվերջ մեծացնել:
- Այդ պատճառով որոնում ենք միայն ուղղությունը, ոչ երկարությունը:

Մի քանի գլխավոր բաղադրիչ

PC2-ը մեծագույն դիսպերսիա ունի PC1-ին ուղղահայաց ուղղությունների մեջ



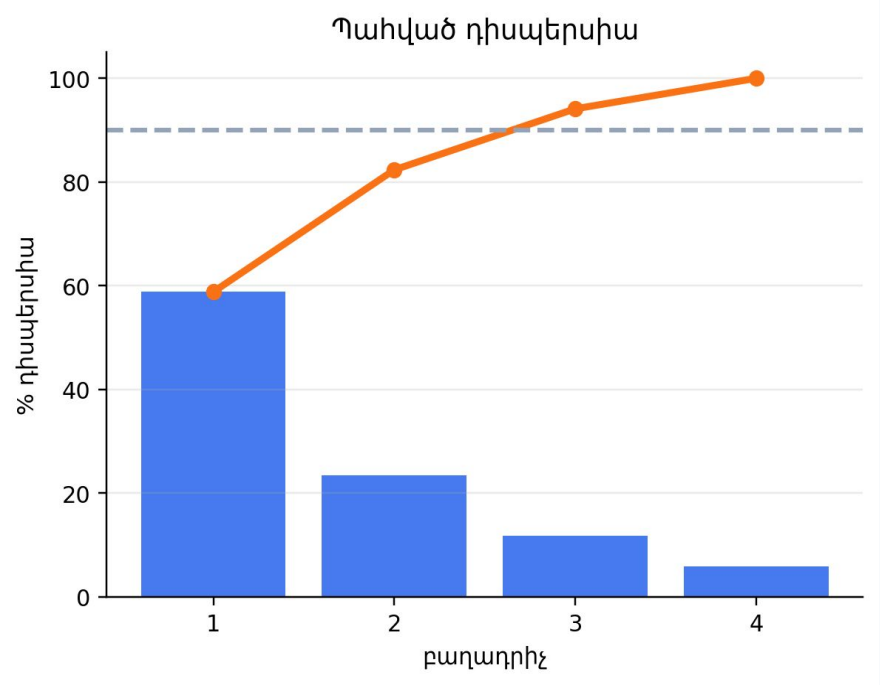
$$W_{\square} = [v_1, v_2, \ldots, v_{\square}]$$

$$Z = X_c W_{\square}$$

- $v_1, v_2, \ldots$ սեփական վեկտորներն են՝ դասավորված մեծից փոքր λ-ով:
- Բաղադրիչները ուղղահայաց են և չկոռելացված:
- Նշանի փոփոխությունը չի փոխում PCA-ի իմաստը:

Որքա՞ն դիսպերսիա ենք պահում

Explained variance ratio-ը ցույց է տալիս պահված տեղեկության բաժինը PCA-ի չափանիշով



$$EVR_r = \lambda_r / \Sigma\lambda_r$$

- Կուտակային գիծը ցույց է տալիս՝ առաջին k բաղադրիչները միասին որքան դիսպերսիա են պահում:
- 90% շեմը կանոն է, ոչ օրենք:
- Կանխատեսման խնդրում k-ն ընտրում ենք validation-ով:

PCA-ի վեց քայլը

Ալգորիթմը, որը հիմա արդեն կարող ենք հասկանալ

1. Կենտրոնացնել տվյալները
2. Քննարկել սանդղակը / ստանդարտացումը
3. Հաշվել կովարիացիայի մատրիցը C
4. Գտնել C-ի սեփական արժեքներն ու վեկտորները
5. Ընտրել k բաղադրիչ
6. Պրոյեկտել՝ $Z = X_c W$

Քայլ 1. կենտրոնացում

Յուրաքանչյուր հատկանիշից հանում ենք իր միջինը

$$\mu_j = (1/n)\ \Sigma_i\ X_{ij}$$

միջին կետը դառնում է սկզբնակետ

$$Xc_{ij} = X_{ij} - \mu_j$$

PCA-ի ենթատարածքը անցնում է տվյալների միջինով

• Եթե չենք կենտրոնացնում, առաջին առանցքը կարող է նկարագրել ոչ թե փոփոխությունը, այլ տվյալների միջինի հեռավորությունը զրոյից:

Քայլ 2. կովարիացիայի մատրից

Հաշվում ենք՝ որ հատկանիշներն են միասին շարժվում

$$C = X_c^T X_c / (n - 1)$$

C-ի չափը d × d է

C-ն սիմետրիկ է

սեփական արժեքները ≥ 0

- Անկյունագծում դիսպերսիաներն են:
- Անկյունագծից դուրս կովարիացիաներն են:
- Դրական կովարիացիան նշանակում է՝ մեծ արժեքները հաճախ միասին են:

Քայլ 3. սեփական տարրալուծում

Գտնում ենք C-ի հատուկ ուղղությունները

$$C\,v_i = \lambda_i\,v_i$$

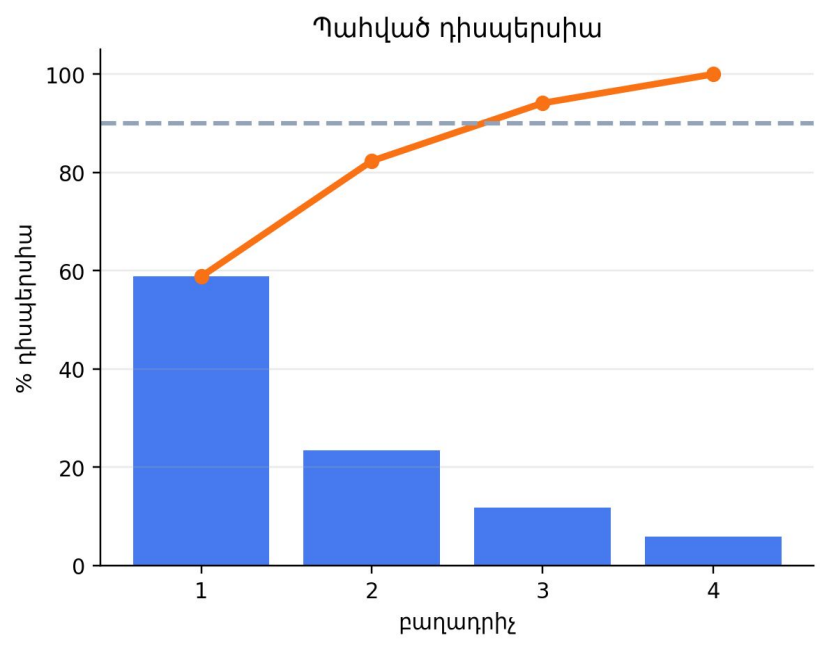
v_i՝ PCA առանցք

λ_i՝ այդ առանցքի դիսպերսիա

- Սեփական արժեքները դասավորում ենք մեծից փոքր:
- Առաջին սեփական վեկտորը PC1-ն է:
- Երկրորդը PC2-ն է և ուղղահայաց է առաջինին:

Քայլ 4. ընտրել k բաղադրիչ

Կրճատման ուժը վերահսկում է k թիվը



- Փոքր k → ուժեղ սեղմում, բայց ավելի մեծ կորուստ:
- Մեծ k → ավելի շատ տեղեկություն, բայց ավելի քիչ կրճատում:
- Ընտրությունը կախված է նպատակներից՝ visualization, compression կամ prediction:

$$k < d$$

Քայլ 5. պրոյեկտել և ստանալ նոր աղյուսակ

Հին հատկանիշներից ստեղծում ենք նոր կոորդինատներ

$$Z = X_c W_k$$

X_c` n × d → $W_k$` d × k → Z` n × k

- Z-ի տողերը նույն նմուշներն են, բայց նոր կոորդինատներով:
- Այս կոորդինատները կոչվում են PCA scores:

NumPy. կառուցում ենք PCA-ն

Նույն քայլերը կոդով

```
import numpy as np

X = np.array([[ 2,  2], [-2, -2], [1, -1], [-1, 1]],
dtype=float)
mu = X.mean(axis=0)
Xc = X - mu
C = Xc.T @ Xc / (len(X) - 1)

values, vectors = np.linalg.eigh(C)   # C- ն սիմետրիկ է
order = values.argsort()[::-1]
values = values[order]
vectors = vectors[:, order]

print(values)
print(vectors)
```

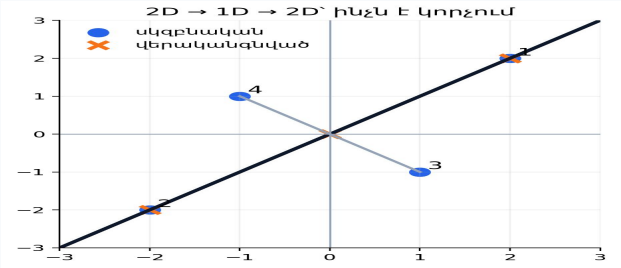
- axis=0 նշանակում է՝ միջինը հաշվել սյուներով:
- @ նշանը մատրիցների բազմապատկում է:
- eigh-ը հարմար է սիմետրիկ մատրիցների համար:

NumPy. պրոյեկցիա և վերականգնում

Պահում ենք միայն առաջին բաղադրիչը

```
k = 1
W = vectors[:, :k]
Z = Xc @ W
X_hat = Z @ W.T + mu

print('Z =')
print(Z.round(3))
print('reconstruction =')
print(X_hat.round(3))

print(np.allclose(W.T @ W, np.eye(k)))
```

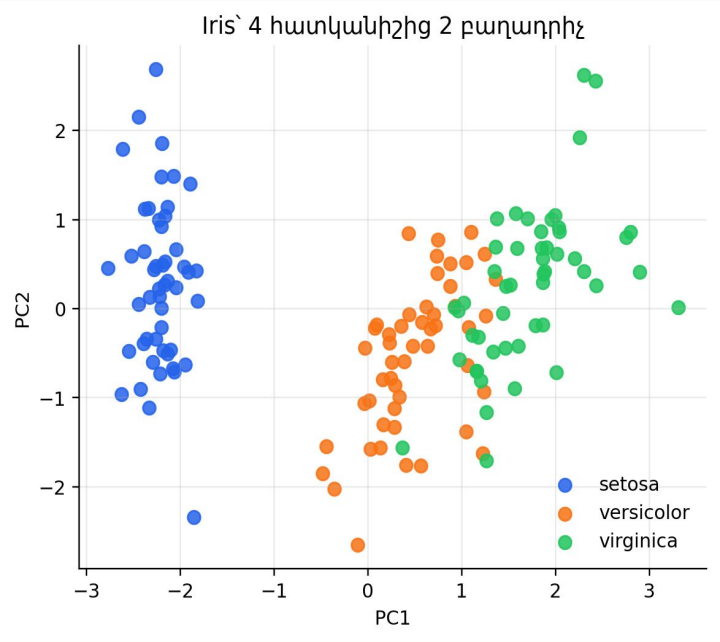
- W-ի ձևը d×k է:
- Z-ի ձևը n×k է:
- X_hat-ը մոտավոր վերադարձ է սկզբնական տարածք:
- Պահել պետք է μ-ն և W-ն:



scikit-learn. Iris տվյալները

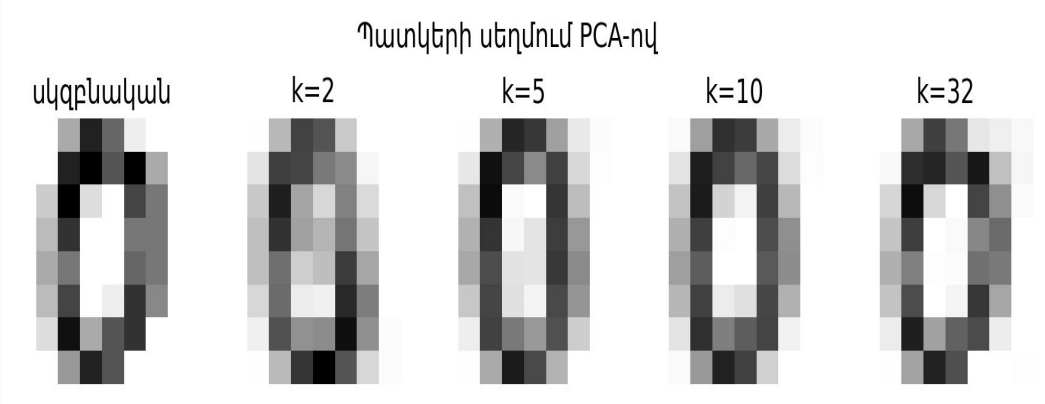
4 հատկանիշից ստանում ենք 2 բաղադրիչ՝ նկարելու համար

```
from sklearn.datasets import load_iris
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA

iris = load_iris()
X, y = iris.data, iris.target

X_scaled = StandardScaler().fit_transform(X)
Z = PCA(n_components=2).fit_transform(X_scaled)

print(Z.shape)  # (150, 2)
```



Պատկերի սեղմում PCA-ով

Քիչ բաղադրիչներ → ավելի մշուշոտ, շատ բաղադրիչներ → ավելի մոտ վերականգնում

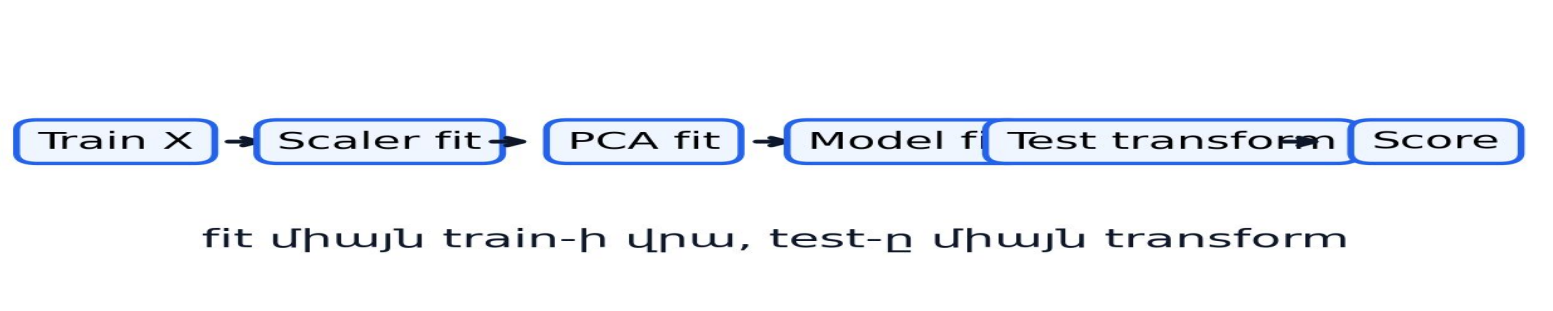


64 → k (այս օրինակում 8×8 պատկեր)

- Յուրաքանչյուր պատկերը կարող ենք ներկայացնել k թվերով:
- Վերականգնված պատկերը նորից ունի նույն պիքսելների թիվը, բայց մանրամասները կարող են կորել:
- Պետք է պահել նաև միջինը և PCA բազիսը:

Կանխատեսման ժամանակ՝ կանխեք տվյալների արտահոսքը

PCA-ն նույնպես սովորում է պարամետրեր



- Train/test բաժանումը արեք մինչև scaler/PCA fit-ը:
- Scaler-ը և PCA-ն fit արեք միայն train-ի վրա:
- Test-ի վրա միայն transform և վերջնական գնահատում:

Ե՞րբ PCA-ն բավարար չէ

PCA-ն հզոր է, բայց ունի սահմաններ

Գծային մեթոդ է

զգայուն է սանդղակին

արտանետումները կարող են շեղել առանցքները

- Եթե տվյալները կորացած մակերևույթի վրա են, PCA-ն կարող է վատ ներկայացնել կառուցվածքը:
- Միայն visualization-ի համար հաճախ օգտագործվում են t-SNE կամ UMAP, բայց դրանք այլ նպատակ և այլ սահմանափակումներ ունեն:
- Եթե պետք է դասերը լավ բաժանել, PCA-ն միշտ լավագույն ընտրությունը չէ:

PCA = լավ առաջին քայլ, ոչ միշտ վերջնական լուծում

Ամփոփում և ելքի հարցեր

Դասի վերջում սովորողը պետք է կարողանա բացատրել այս 4 միտքը

1. Ո՞րն է տարբերությունը ռեգրեսիայի և դասակարգման միջև

2. Ինչո՞ւ է PCA-ն չվերահսկվող մեթոդ

3. Ինչպե՞ս է կովարիացիայի մատրիցը կապված PCA առանցքների հետ

4. Ի՞նչ է նշանակում “պահեցինք 90% դիսպերսիա”

Մեծ գաղափար՝ շատ հատկանիշ → քիչ, իմաստալից կոորդինատ