Object Linear Algebra for AI (شرح بالعربى) Date / /

فيديو 1 :- الدقة for
عندى صورة الـ Image ابعادها (912*621) عاوز اعملها ضغط علشان الـ [illegible] ماتاخدش وقت و[illegible]
أشكال الـ [illegible] والـ size ميكونش كبير.
عملت [illegible] الى (912*2) ⇐ ازاى بقا أعرف الـ maximum number of columns
اللى وخدت بعدية مش هيبقى عندى معلومات ~~[illegible]~~ زيادة. ([illegible] الـ columns اللى وخدتها هيبقى
نفس الـ [illegible] الأصلية بالضبط والباقى ملوش لازمة عندى).
فاذا حصل دقة هنا دى (912*390) اللى زودت عليه مش هيزود حاجه.
عرفنا بقا منين الرقم ده. جبناه اسمها (Image Matrix Rank).
⇐ بناء على العلاقة بين الـ features x , الـ labels y بنحدد نوع الـ model المستخدمه
بنحاول نستوفى الـ General model بينفع لأى observations جيه التحليل الـ error بين
الـ $\hat{y}$ و الـ y اقل ما يمكن. بدأ يتكلم على معادلة عامة تطبق الـ [illegible] observations ويبدأ
بيقول لو معانا معادلة عامة عبارة عن General $Y = \theta_1 x_1 + \theta_2 x_2$

x_1	x_2	y
4	5	11
2	3	8

$$\rightarrow \begin{cases} 4\theta_1 + 5\theta_2 = 11 \\ 2\theta_1 + 3\theta_2 = 8 \end{cases} \rightarrow \begin{matrix} \theta_1 ✓ \\ \theta_2 ✓ \end{matrix}$$

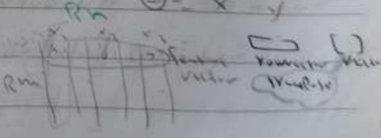
ومكن لو زودت observations هيكون عندى ٣ معادلات فى مجهولين.
ولو حذفت observations واحدة هيكون عندى معادلة واحدة فى مجهولين.

لازم تكون معطوفة مربعة [illegible]
[illegible]

System of linear equation

$$\underbrace{\begin{bmatrix} 4 & 5 \\ 2 & 3 \end{bmatrix}}_{\text{matrix}}^{X} \underbrace{\begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix}}_{\text{vector}} = \underbrace{\begin{bmatrix} 11 \\ 8 \end{bmatrix}}_{\text{vector}}^{Y} \qquad \rightarrow X^{-1} X \theta = X^{-1} Y \qquad \theta = X^{-1} Y$$

What is vector? From (CS, DS) → List of number



عن دقة اعتبر كل column ⇐ vector و كل row ⇐ vector برده والـ [illegible] ?
From LA → الـ vector طالع من الـ origin [illegible]
[illegible] كلها vectors من نقطة الأصل فبنتعامل مع النقطة كأنها vector

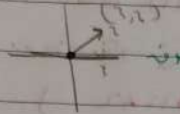
Date / /

ميدو؟ دايماً بنتعامل مع الـ vector في الـ (cartesian)

على حسب عدد الـ (dimensions) بيكون الـ vector أقدر أحطه

نشوفه في dimensions

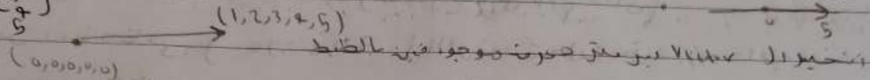
$V = \begin{bmatrix}3\\2\end{bmatrix} \rightarrow$



$V = [5] \rightarrow$ vector in one dimension

مفهوش فرق بين الشكلين وده الـ Scalar

$V = \begin{bmatrix}1\\2\\3\\4\\5\end{bmatrix}$



ضروري نحيل الـ vector دي بيعتبر صورة موجودة في الطبيعة

مهما كان عدد الـ dimensions دايماً بيبدأ من خارج من الصفر رايح للنقطة

magnitude → | |

$V = \begin{bmatrix}4\\3\end{bmatrix} \rightarrow \overset{\text{Scalar}}{2}V = \begin{bmatrix}8\\6\end{bmatrix}$ (بيحصل scaling بالزيادة) الطول زاد الضعف

ممكن يكون ده في direction عكسه

Vector addition $V = \begin{bmatrix}1\\2\end{bmatrix}$ $U = \begin{bmatrix}2\\3\end{bmatrix}$

$W = V + U = \begin{bmatrix}3\\5\end{bmatrix}$

$W_1 = \underset{\text{Scalar multiplication}}{2V} + U$ vector addition ; $W_2 = 2V - 2U$

عندنا V و U فلما نعمل لهم ... نقدر نعرف الـ $W_1, W_2, W_3, \dots$ بدلالة الـ U والـ V الوحيدة

$\rightarrow w = aV + bU$ ← لو عملنا الجمع غير الضرب عادي لاحظ دول العمليات الوحيدة

in linear Algebra ⊕ ← scalar

اللي بنطبقهم على الـ vector اللي بنسميها linear (خارجين من نقطة الأصل رايح للنقطة)

Linear Algebra :- $\boxed{\text{linear transformation}}$ والـ vectors العلم المسؤول عن دراسة الـ

vector norm $\|V\|$ → magnitude طول الـ vector → Distance measure مقياس مسافة

L1, L2, Lp, maximum norm — norm بيوجد أنواع تانية للـ. (L2 norm) Shortest distance $\sqrt{v_1^2 + v_2^2}$

Ridge of vector ≡ Euclidean distance ≡ Mean square error

MSE بنستخدمه في ... ممكن أوصل للنقطة الناتجة بأكتر من طريقة $\|\vec{v}\|^2$ squared

L_1 norm $= \|V\|_1 = |v_1| + |v_2| + \dots$ (مجموع الـ components) (Manhattan dis) (taxicab distance) (LASSO) (MAE) $\frac{L_1 \text{norm}}{m}$

$L_3 \text{norm} = \left(|v_1|^3 + |v_2|^3 + |v_3|^3 - |v_n|^3\right)^{\frac{1}{3}}$

$L_4 \text{norm} = \left(|v_1|^4 + \dots + |v_n|^4\right)^{\frac{1}{4}}$

L_∞ norm = max أكبر قيمة في الـ vector |max| قيمة كبرى

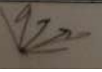
Largest component

بتعلمنا أجمع vector و أطرح لازم يكون لهم نفس الـ dimensions

في python من الـ library اسمها math

نجيب حاجة اسمها (numpy)

* ممكن حاجة اسمها ... ممكن تكون لازم يتطلعوا من نفس النقطة



مجموعة أرقام من الـ start والـ end الشغال

Object Date / /

Dot Product $V_1=\begin{bmatrix}5\\2\end{bmatrix}$ $V_2=\begin{bmatrix}2\\3\end{bmatrix}$ $V_1\cdot V_2=$ Scalar سيبو ٣

$|| \ ||\cos\theta$

معنى ال dot product ⇐ $|v_1| \ |v_2|\cos\theta$ ⇐

اسقاط المتجه على V_1 → $\boxed{\text{norm}\cos\theta}$ norm لو كانت unit vector

$|v_1| \ |v_2|\cos\theta$

dot Product = Projection الاسقاط حسبنا منها ← $|v_1| \ |v_2|\cos\theta$ norm

من اهم استخدامات projection هو ال dot product لو عندنا متجه unit vector

حاب متجه ونعبر عنه بدلالة متجهات ثانية $b=\begin{bmatrix}4\\2\end{bmatrix}$ → $2v_1+2v_2$

v_1+v_2 / $2v_1+2v_2$

بس المهم ان b متغيرش [ال reference المتغير منه هو التعبير ال b واقفة في مكانها)

(اتذكر لما b ثابتة واقفة في مكانها و بنوصف الواحد بوصلة ازاى) ال Pattern بيهمس كما بعد غير متحرك

x_1	y_1
5	2
4	3

⇒

z_1	z_2
2	0.5
1.5	2.5

لانها متغيرتش ولكن اتحولت نتيجة انك غيرت ال reference المستخدمة هنا. (مثل ما ان الارقام اتغيرت بس الحاجة ثابتة)

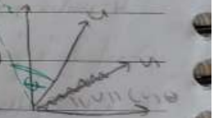
← نقدر نعرف المتجه بدلالة ال dot product

علشان نجيب ال projection v_2 $u\cdot v_1=||v_1|| \ ||u||\cos\theta$

لو ال v لاليس unit vector وبينهم زاوية $||u||\cos\theta=\frac{u\cdot v_1}{||v_1||}$

بس ان ال $||u||\cos\theta$ جزء من الطول كله بتاع ال v_1 $\frac{||u||\cos\theta}{||v_1||}=\frac{u\cdot v_1}{||v_1||^2}=a$

لاستخدمه علشان نكون النسبة من الكل ثابت



عاوز نكتب ال u بدلالة ال v_1, v_2 هنحسب من خلال ال dot product وال unit vector $\bar{v}_1$ ⓐ = u_{v_1}

بدلالة المتجهات v_1+v_2

ال a الجزء نسبة الى الطول الكلي الطول ⓐ

$u_{v_2}=b \ \bar{v}_2$ ∴ $u=av_1+bv_2$

$b=\frac{u\cdot v_2}{||v_2||^2}$

لو عندنا v_1, v_2 (متعامدين لازم) نقدر نعبر عن ال u بدلالة ال v_1 و ال v_2 من غير الزاوية

$$u=\frac{u\cdot v_1}{||v_1||^2}\bar{v}_1+\frac{u\cdot v_2}{||v_2||^2}\bar{v}_2$$

بعدت الى ان المتجهين يكونوا $||v_1||^2=v_1\cdot v_1$ $||v_2||^2=v_2\cdot v_2$

متعامدين انهم عاديين ما يعبر ولا يبين لازم $u=\frac{u\cdot v_1}{v_1\cdot v_1}\bar{v}_1+\frac{u\cdot v_2}{v_2\cdot v_2}\bar{v}_2$

(يكونو اعبر لاخذ في المحور بتاع (المقابلة) ال سهولة حساب القيمة بسيطا في المقابلة

لو غير متعامدين نستخدم في حسابات متعقدة

Matrices $\overset{\text{linear}}{5x + 6y = 3}$

linear → $2x + 5y = 1 \Rightarrow \overset{\text{matrix} \times \text{vector}}{\begin{bmatrix} 5 & 6 \\ 2 & 5 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}} = \begin{bmatrix} 3 \\ 1 \end{bmatrix}$

linear function applied to the vector $\{x, y\}$

$\{x, y\}$ بضربه في ال matrix ده وحولته ل $\{3, 1\}$

التعبير ال $\{x, y\}$ الما اضرب في ال linear function بتحول الى $\{3, 1\}$

ح linear function بسموها linear transformation لانها عملت transform (matrix) للمتكورة من $\{x, y\} \rightarrow \{3, 1\}$

Operations on Matrices

الجمع والطرح لازم نفس ال dimension

ضرب AB ≠ BA

$(AB)^{-1} = B^{-1}A^{-1}$ نفس الكلام ال inverse برده — Transpose ← $(AB)^T = B^T A^T$

Different types of Matrices

1) Identity Matrix → square matrix the diagonal = 1 $\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ I

لو ضربته في اي مصفوفة بترجع زيها لا تؤثر على اي حاجة

2) Diagonal Matrix → زي ال Identity بس عناصر ال diagonal بأرقام عادية

← لو ضربتها في نفسها ولنفرض 100 مرة فتعتبر الناتج نفس المصفوفة بس عناصر ال diagonal مرفوعة لأس المرات دة زي

$A = \begin{bmatrix} 3 & 0 & 0 \\ 0 & 8 & 0 \\ 0 & 0 & 1 \end{bmatrix} \rightarrow (A)^{100} = \begin{bmatrix} 3^{100} & 0 & 0 \\ 0 & 8^{100} & 0 \\ 0 & 0 & 1^{100} \end{bmatrix}$

وده مهمة عشان لو كانت مصفوفة عادية هتبقى عملية مرهقة جدا للمعالج كمان

3) Scalar matrix $\begin{bmatrix} 5 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & 5 \end{bmatrix}$ لها نفس عناصر ال diagonal

لو ضربتها في اي matrix تانية هتاخد كل ال matrix تضربه في ال scalar ده

$5 * \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$

4) Lower triangular matrix (L) تحت ال diagonal ارقام فوق ال diagonal اصفار

بيساعدني في حل ال equations (Forward substitution) $\begin{bmatrix} 5 & 0 & 0 \\ 2 & 1 & 0 \\ 7 & 6 & -3 \end{bmatrix}$

5) Upper triangular matrix (U) الفوق ارقام والتحت باصفار بيقدر احل

المعادلات بطريقة بسهولة (Back substitution)

6) Symmetric matrix $\begin{bmatrix} 3 & -1 & 11 & 5 \\ -1 & & & 6 \\ 11 & & & 7 \\ 5 & 6 & 7 & 9 \end{bmatrix}$ متماثلة حول ال diagonal بيساعدني بسرعة

A = A^T (1 Save memory) اقدر اخزن نصها بس (2 بيعملوا حاجة اسمها decomposition

بتتحول لـ (orthonormal eigenvectors and diagonal matrix of eigenvalues)

Object Date / /

non zero اول   تحته اصفار   pivot

7) Row echelon Form (REF) $\rightarrow \begin{bmatrix} \boxed{5} & 6 & 7 \\ 2 & 2 & 2 \\ 6 & 6 & 6 \end{bmatrix} \xrightarrow{REF} \begin{bmatrix} \boxed{5} & 2 & 1 \\ 0 & \boxed{1} & 2 \\ 0 & 0 & \boxed{3} \end{bmatrix}$

تحته اصفار

اول non zero تحته اصفار اة (REF) هي زي ال upper لو كانت square

ممكن يجب REF الـ non square حالي ولكن لازم square

8) RREF , هتعمل لل REF ← reduction لازم ال pivot تحته اصفار

والباقي 1 تحته وفوقه اصفار $\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

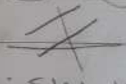
والمفروض نعرف ازاي نشكل وازاي نفرق REF و RREF

Tensor: عبارة عن matrices وبعضهم فوق بعض في ال image بتخزن في 3D tensor

System of linear equation and How to solve

$x - 2y = 1$

$3x - 6y = 11$



نفس الميل ولكن متغير التقاطع فهيكون خطين متوازيين ليس لهم حل

$\begin{bmatrix} 1 & -2 \\ 3 & -6 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 1 \\ 11 \end{bmatrix}$

$A \quad x = b$

A ← حالة نعتبر solution مش موجود حل A^{-1}

$x - 2y$ هم تكرار redundant features عندك

فالماتريكس هنا يبقى بعض vector (x, y) نضرب ايه له matrix ال $\begin{bmatrix} 1 & -2 \\ 3 & -6 \end{bmatrix}$ يحوله الي $\begin{bmatrix} 1 \\ 11 \end{bmatrix}$

Problem 2 $x - 2y = 1$

$3x - 6y = 3$ عدد لانهائي من الحلول

$\begin{bmatrix} 1 & -2 \\ 3 & -6 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 1 \\ 3 \end{bmatrix}$ عن عدد لانهائي من ال vector $\begin{bmatrix} x \\ y \end{bmatrix}$ لو ضربتهم في المصفوفة $\begin{bmatrix} 1 & -2 \\ 3 & -6 \end{bmatrix}$ تحولها الي vector $\begin{bmatrix} 1 \\ 3 \end{bmatrix}$

Problem 3 $x - 2y = 1$

$3x + 2y = 11$

$\begin{bmatrix} 1 & -2 \\ 3 & 2 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 1 \\ 11 \end{bmatrix}$ ⇐ حل وحيد

يعني هناك vector واحد بس ( ) لو ضربته في ال matrix $\begin{bmatrix} 1 & -2 \\ 3 & 2 \end{bmatrix}$ هيدي ال vector $\begin{bmatrix} 1 \\ 11 \end{bmatrix}$

Object ولو matrix المطرح
طلعت في عمور ميتفعش عمود صفر (V1@V2) dot product Date / /

Ex: $\begin{bmatrix}1 & -2\\ 3 & 2\end{bmatrix}\begin{bmatrix}x\\ y\end{bmatrix}=\begin{bmatrix}1\\ 11\end{bmatrix}$ لو هنعمل عمليات عليه بنشيلة وبنفرض التالي

Gaussian elimination

بيستخدم لحساب التحتي تكون صفر

$\begin{bmatrix}\boxed{1} & -2 & : & 1\\ 3 & 2 & : & 11\end{bmatrix} \rightarrow \begin{bmatrix}1 & -2 & | & 1\\ 0 & 8 & | & 8\end{bmatrix}$ REF $\rightarrow \begin{bmatrix}1 & -2\\ 0 & 8\end{bmatrix}\begin{bmatrix}x\\ y\end{bmatrix}=\begin{bmatrix}1\\ 8\end{bmatrix}$

Back substitution التعويض

$$\begin{bmatrix}2 & 4 & -2\\ 4 & 9 & -3\\ -2 & -3 & 7\end{bmatrix}\begin{bmatrix}x\\ y\\ z\end{bmatrix}=\begin{bmatrix}2\\ 8\\ 10\end{bmatrix}$$

$$\left[\begin{array}{ccc|c}2 & 4 & -2 & 2\\ \boxed{4} & 9 & -3 & 8\\ -2 & -3 & 7 & 10\end{array}\right] \xrightarrow{①} \left[\begin{array}{ccc|c}② & 4 & -2 & 2\\ 0 & 1 & 1 & 4\\ -2 & -3 & 7 & 10\end{array}\right] \xrightarrow{②} \left[\begin{array}{ccc|c}\boxed{2} & 4 & -2 & 2\\ 0 & \boxed{1} & 1 & 4\\ 0 & 1 & 5 & 12\end{array}\right]$$

$$\xrightarrow{③} \left[\begin{array}{ccc|c}2 & 4 & -2 & 2\\ 0 & 1 & 1 & 4\\ 0 & 0 & 4 & 8\end{array}\right] \rightarrow \left[\begin{array}{ccc|c}2 & 4 & -2 & 2\\ 0 & 1 & 1 & 4\\ 0 & 0 & 1 & 2\end{array}\right] \rightarrow \left[\begin{array}{ccc|c}2 & 4 & -2 & 2\\ 0 & 1 & 0 & 2\\ 0 & 0 & 1 & 2\end{array}\right]$$

$$\left[\begin{array}{ccc|c}2 & 4 & 0 & 6\\ 0 & 1 & 0 & 2\\ 0 & 0 & 1 & 2\end{array}\right] \rightarrow \left[\begin{array}{ccc|c}1 & ② & 0 & 3\\ 0 & 1 & 0 & 2\\ 0 & 0 & 1 & 2\end{array}\right] \rightarrow \left[\begin{array}{ccc|c}1 & 0 & 0 & -1\\ 0 & 1 & 0 & 2\\ 0 & 0 & 1 & 2\end{array}\right]$$

x = -1
y = 2
z = 2

الخطوة اللي بتكمل الـ Gaussian elimination ← اللي تحول الـ matrix إلى identity

اسمها Gauss Jordan method

V4-20V5

ازاي نتعامل مع الـ numpy - لو هنتعامل vector مع matrix [[2]] [2]   numpy. الـ vector بيبقى matrix

مش هيتكتب بالشكل ده هيتكتب بشكل تاني

( ) @ ( ) ✓

[ ] @ ( ) ✓

[ ] @ [ ] ×

بنتعامل مع الـ 1D Array عادي ونشيل اما الـ 2D Array فيها صرامة

ولازم الـ @ يكون عدد الأعمدة الأولى في صفوف الثانية

كتابة matrix في [illegible] matrix

أو نكتبها كاملة بنفضل

`import sympy as sm` →

`sm.Matrix( )` → دي مش لازم حتى الـ numpy

فنحول الـ array numpy الى matrix

`Aug.rref( )` يحلل المعادلة البسيطة Gauss Jordan

لحد امور من خلال الـ sympy نستعمل كلهم النتايج
`x, y = sm.symbols('x y')`

`sm.linsolve(Augmented, ( ))`

المستعمل كلهم النتايج

index for pivots → الموجودة

Object Date / /

هنبدأ بنستخدم الـ library ndarray → object numpy

لو عندنا [1, 2, 3] → الشكل ده لا row vector ولا column vector v1=np.array([المحتوى])

لإنك بيعتبروه مش محدد، إنت عاوزه ممكن يكون row / vector لها ميزة مثلا لو عاوز نعمل

[illegible] على الشكل ده عادي و واحد يعتبرو row و واحد يعتبرو column بـ np.dot

v1.reshape((,))
لو عاوز تغير الـ shape

الـ dot product لـ array يعني array مش scalar زي الـ vectors لو عاوز

أول عنصر من الـ array يعني (0,0) وكذا ...

← نعمل العمليات الخاصة بالـ matrices والـ vector أو row و column وده عن طريق reshape

arr.T ✓

np.transpose(arr) ✓

np.linalg.inv(arr)

np.linalg.det()
محدد

arr[0,1] → أول رقم

arr[0] → الصف كله

arr.sum() → بجمع كل الأرقام في المصفوفة

عاوز أجمع كل عمود لوحده لو شغال رأسي axis=0

عاوز أجمع كل صف axis=1

نتيجة الـ [illegible] shape يكون مش محدد مثلا axis=0 أو axis=1

خلي الـ keepdims=True

v1.reshape(-1,1) ← تراجعها
عدد الصفوف كلها
الـ dimension

np.zeros(( ))

np.eye(n)

np.ones(( ))

np.arange(1,100,5)

ممكن أعمل [illegible] على المصفوفة ماشي arr[arr>20]

np.linspace(0,1,n)
uniform distribution

المصفوفة نهايتها عبارة عن أرقام كسور

np.random.rand(4,3) 0→1

لكل عنصر ← numpy random uniform

randn
normal distribution
m=0
v=1

np.random.randint(0,100,20)

arr.shape (2,3)

arr.size
6 عدد العناصر

Object [vector space - linear combination - vector span - linear independence - vector basis] Date / /

R1

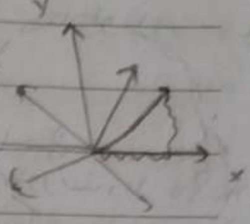
1) Vector space → هنبدأ 5 أهم دروس.. ال space اللي بيحتوي على كل ال vectors في ال dimension دة محتاج بس علشان نمثلة ال 2 dimension

component

V ∈ R^2 space ← ال 2 vectors اقدر اجمعهم و scalar multiplication $V = \begin{bmatrix}3\\1\end{bmatrix}$

العمليات اللي اقدر نعملهم بسرعة

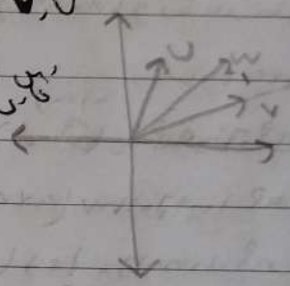
نفس دة ممكن يحصل في (R^1, R^3, ... R^n)



2) Linear combination → دي حاجة متحصرة لل R^2 هي ال R^2 في ال R^3

تجميع من دول

$w_1 = v + u$	w_1 is a linear combination from v, u
$w_2 = 2v + u$	w_2 is . . . . . . بس مختلفين الأوزان
$w_3 = 2v + 2u$	w_3 . . . . . .
⋮	



$w_1 = \overset{1}{a}v + \underset{1}{b}u \quad w_2 = \overset{2}{a}v + \overset{1}{b}u \quad w_3 = \overset{2}{a}v + \overset{2}{b}u$

uv vv

← لو سألتك لو لقيت متجه معين هل هو linear combination بين متجهين تانيين ولا لا هتعرف ازاي؟

← هنعمل check كدة

$$\overset{w_1}{\begin{bmatrix}3\\5\end{bmatrix}} = a\overset{v}{\begin{bmatrix}1,5\\0,3\end{bmatrix}} + b\overset{u}{\begin{bmatrix}-0,1\\-4\end{bmatrix}}$$

لو عرفت أجيب قيمة لل a, b يبقى فعلا ال w_1 هيكون linear combination من v, u

هنعمل system of linear equation بين ال (v,u) مع ال a و b ونحسبه ال (a,b) بحيث يكون المتجه الناتج هو ال w_1

$$\begin{bmatrix}v & u\\1,5 & -0,1\\0,3 & -4\end{bmatrix}\begin{bmatrix}a\\b\end{bmatrix} = \begin{bmatrix}3\\5\end{bmatrix}$$

ملحوظة ← الحل بتاع ال inverse لو عملنو لو المسألة ليها حل وحيد هيشتغل معاك أما لو غير كدة مش هيشتغل عشان كدة بنستخدم ال RREF الموجود

← أي vector في ال R^2 نقدر نعبر عنه بدلالة متجهين ولكن مش هينفع من خلال متجه واحد بس الا لو على نفس الخط بتاعه

← ممكن نعمل لـ vector و نسوقه من خلال متجهين بس مساحات تنفع ومساحات متنفعش
1D 3D أما لو متجهين في ال 2D زي ما قلنا لازم متجهين

Object Date / /

طيب يعني هي الـ 3 vectors دول ممكن يكونوا متعامدين او في نفس الـ plane !!

لو المتجهة الثالث اعملو لمتجهين بيعملو زاوية الـ plane المحصور الـ 2 vectors التانيين دول

لو كان الـ w دة بين الاتنين في الـ Plane الذي يوجد فيه الـ 2 vectors فممكن

← علشان اعمل متجهة في 3D لازم يكون عندي 3 بعد من كل dimension (x, y, z)

V1 الـ ... v1, v2, v3

span of V1 ← كل الـ vectors القدر اكتبهم كـ linear combination من الـ V1

اقدر اكتبه بدلالة الـ V1 عدد لا نهائي من الـ vectors على نفس الـ line

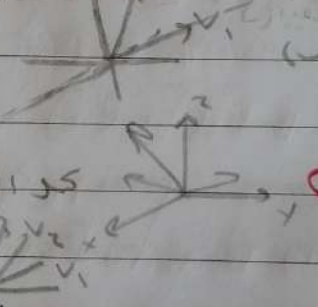
span(V1)

الـ span هي كل الـ vectors القدر اعبر عنها من خلال الـ V1 و على نفس الخط بتاعة الخط المستقيم لو وصلت النقط ببعضها هو الـ span

طب لو R² ؟ ← بيـ span ← 1 dimension subspace in R²

هيكون في كل نقطة الـ line

span (V1, V2) ← كل الـ vectors المكتوبة بدلالة V1, V2



طب لو R³ ، كل الـ R³ ...

تخيل عندك 3 vectors V1, V2, V3 نقدر نكتب اي متجهة بدلالة الاتنين التانيين

linearly dependent

aV1 + bV2 + cV3 = 0

dependent vectors

لو مثلا واحد فيهم مثلا V3 = bV2 يعني الـ V1, V2 فيعملو independent vectors

متعامدين نكتب واحد بدلالة التاني . مثلا لاحظ عاوزين نعرف هل الـ 3 vectors

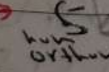
$a\begin{bmatrix} \\ \end{bmatrix} + b\begin{bmatrix} \\ \end{bmatrix} + c\begin{bmatrix} \\ \end{bmatrix} = 0$ V و V dependent

a b c

لو لقيت الـ a, b, c طلعت صفر معنى كدا انهم معتمدين على بعض

المتجهة او المتجهات القدر اكتب بدلالتهم اي متجهة تانية زي كدا

basis →



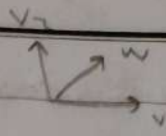
non orthogonal

اقدر اكتب من الـ V1, V2 اي متجهة في الـ 2D

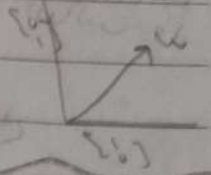
Object

Date / /

orthogonal basis :- special case from Basis



orthonormal → متعامدة ووحدة || || = 1



الصورة محفوظة vectors تحول ل vectors ال

linear transformation → preserve linear combination →

بتشتغل على vector أو مجموعة vectors زي دي منتقول

$$\underbrace{\begin{bmatrix}5&2\\3&4\end{bmatrix}}_{T}\begin{bmatrix}1\\2\end{bmatrix}=\begin{bmatrix}9\\11\end{bmatrix}$$

طب ده بيحصل إزاي ؟؟؟

effect →

العملية على ال $\begin{bmatrix}1\\2\end{bmatrix}$ vector وحولته الى $\begin{bmatrix}9\\11\end{bmatrix}$ vector

طب ليه linear transformation عشان ال T دة linear بتعادلها لأخر معادلات linear function

طالع من ال origin

ك يفضل ال vector برة

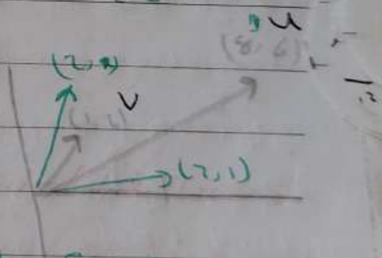
ال vector الناتج من عملية ال transform بيضع عليه أو بيتميز بحاجتين

نقدر نكتبها بالصورة الجاية

$$1\begin{bmatrix}5\\3\end{bmatrix}+2\begin{bmatrix}2\\4\end{bmatrix}=\begin{bmatrix}9\\11\end{bmatrix}$$

L.C from

$$\begin{bmatrix}2&3\\4&1\end{bmatrix}\begin{bmatrix}1\\2\end{bmatrix}=\begin{bmatrix}8\\6\end{bmatrix}$$



$$\begin{bmatrix}8\\6\end{bmatrix}=\boxed{1}\begin{bmatrix}2\\4\end{bmatrix}+\boxed{2}\begin{bmatrix}3\\1\end{bmatrix}=8\bar{e}_1+6\bar{e}_2$$

$$V=1\bar{e}_1+2\bar{e}_2=$$

$$\begin{bmatrix}1\\2\end{bmatrix}=\boxed{1}\begin{bmatrix}1\\0\end{bmatrix}+\boxed{2}\begin{bmatrix}0\\1\end{bmatrix}=1\bar{e}_1+2\bar{e}_2$$

$$\begin{bmatrix}2&3\\4&1\end{bmatrix}\begin{bmatrix}1\\0\end{bmatrix}=\begin{bmatrix}2\\4\end{bmatrix}$$

$$\begin{bmatrix}2&3\\4&1\end{bmatrix}\begin{bmatrix}0\\1\end{bmatrix}=\begin{bmatrix}1\\3\end{bmatrix}$$

نقدر نلاحظ ان ال matrix بتعكس ال effect على ال $\begin{bmatrix}1\\2\end{bmatrix}$ vector لو كتبت ال effect دة على ال basis vectors بتوع ال vectors حتت ال $\begin{bmatrix}1\\0\end{bmatrix}$ $\begin{bmatrix}0\\1\end{bmatrix}$ خلتهم $\begin{bmatrix}2\\4\end{bmatrix}$, $\begin{bmatrix}3\\1\end{bmatrix}$ وهتلاقي ال $\begin{bmatrix}1\\2\end{bmatrix}$ هيكون اتعامل ال basis ال $\begin{bmatrix}1\\0\end{bmatrix}$, $\begin{bmatrix}0\\1\end{bmatrix}$ ال ... هيفضل موجود ال new basis الجديد $\begin{bmatrix}3\\1\end{bmatrix}$, $\begin{bmatrix}2\\4\end{bmatrix}$ فهيدي vector جديد الهو $\begin{bmatrix}1\\2\end{bmatrix}$ بس بدلالة ال new basis ودلوقتي $\begin{bmatrix}2\\4\end{bmatrix}$, $\begin{bmatrix}3\\1\end{bmatrix}$

ال $\begin{bmatrix}8\\6\end{bmatrix}$ لو عاوز اكتبه بدلالة $\bar{e}_1, \bar{e}_2$

ال vectors مع احتفاظ صحة ال basis في
التعديلات البكتب بيشيلوا ال vector

Object changing basis ال ... ذاتي transformation = matrix Date / /

* ال space كله بيعملوا transformation لما نضربها في matrix (ال basis ذات الحلول) transformation (ميزان)

* أنواع ال transformations 1) Scaling 2) Reflection 3) Shearing

4) rotation بيحركوا بنفس الزاوية بالنسبة لبعض بنفس الزاوية

← لو عندك vector عبارة عن مجموع اتنين vectors تانيين وعملت transform للاتنين vectors الموجودة
دول وال vector ده هتلاقي إن ناتج الجمع هو ال transform لل vector ده (بيحافظ على ال L.T)

reverse transformation ⟸

$$\underbrace{\begin{bmatrix}1&2&3\\2&5&2\\6&3&1\end{bmatrix}}_{A}\underbrace{\begin{bmatrix}x\\y\\z\end{bmatrix}}_{X}=\underbrace{\begin{bmatrix}6\\4\\2\end{bmatrix}}_{b}$$

لو معاك إني عاوز أشوف ال vector اللي لما
اضربه في ال matrix ده أتحول إلى ال vector ده ←
يعني هنا ال transformation كان ال vector عبارة
عن x, y, z توصل للمعادلة دي ال system of linear equations. يعني ده معناه إني عاوز ال reverse transformation

$Ax=b \Longrightarrow x=\boxed{A^{-1}}b \rightarrow$ معناه إني بعمل reverse transformation لل vector اللي هو b

طب لو مفيش inverse لل A (A^{-1}) ده معناه إن ال transformation ده (cannot be reversed)

وبردو زي ما كنت بقول إني لما بعمل transformation لـ vector بحوله ل basis تانية، ال vector space

لما بعمل reverse transformation بترجع كل ال vectors لأصلها ما قبل ال transformation

معنى ده إني أعرف هل ال transformation ده invertible ← فيه حل / not invertible ← موجود حلول لانهائية أو مفيش حل

إيه معنى المحدد للمصفوفة أصلاً!!!!؟ Matrix Determinant →

ال Determinant بتاع أي transformation matrix بيمثل (constant ← determinant) $\boxed{A}$ ...

ال area بتاعة ال space هيحصلها scaling بمقدار مساحة ال space
الفيكتورز (vectors) ال constant هو ال determinant

خلي بالك إن ال area دي ال space كلها هيحصلها نفس ال constant ده

ما معنى ال det بالسالب ← علامة ال (-) يعني ال axis بيعكس، يعني انعكست زي لو قلبتها ال
plane كلها.

طب في 3 dimensions ده هيكون عبارة عن volume ونفس الحاجة determinant ده V ... 3-D

ما معنى إن ال det = 0 ← يعني 0.A = 0 المساحة بقت صفر (ال vectors بقوا على نفس الخط)

$\begin{bmatrix}4&2\\2&1\end{bmatrix}$ linear dependent vectors

خلت كل ال vectors الموجودة في ال space بقت على line واحد

لما تقلبها ال space كلها بقت line، أنا مش هقدر أرجع كل vectors وأشوف ال vector كان إيه؟

أكيد لا ده ال transformation ده (cannot be restored)

في ال 3 dimension يا إما (plane) أو line أو space

det=0 → singular matrix → (cannot be reversed)

عدد الصفوف الـ independent هي عدد الأعمدة الـ independent حتى لو مش square (ميكونش فيه حاجات زيادة)

*لو الـ matrix فيها صف كامل أصفار الـ det = 0

Object Row space & column space always consistent　　Date / /

$x-2y=1$ 　 $x-2y=1$

$3x-6y=3$ 　 $3x-6y=11$

لو الـ equations موجودة في نفس الـ line هيكون عندي عدد لا نهائي من الحلول نفس الـ vectors يعتبر line واحد، أما لو الـ equations بيمثلوا line مستحيل يكون فيه حل

Matrix rank :- 　 1 → 1 independent يكون عدد الـ independent اللي هنا علشان

2 → 2 ... أخرى هو ... الـ columns

3 → 3 ... independent columns كام

rref ← يبطل pivots ← عدد الـ pivots في الـ rank كذا

الـ nonsquare ← الـ rank لا يزيد عن عدد الـ columns أو عدد الـ rows

الـ rank عبارة عن كمية الـ vectors الـ independent in matrix كام واحد independent و dependent

(CF A) → 18 → قللت الـ size عدد الـ ... 18 ← 20 صورة مثلا لو عدد الـ vectors كثير جدا

Matrix Factorization:- $A \Rightarrow \boxed{CF}$ تحولها لحاصل ضرب اتنين أو 3 matrices

Matrix (Decomposition)

فيها خواص نقدر نستفيد منها من الـ matrices ...

في (Rank Factorization , QR , SVD , LU , ...)

أنواع من تقسيم الـ matrix الأصلية لعدة vectors

A 4×5 + first 2 columns independent and 3 after are dependent

يعني المعلومات المفيدة موجودة في أول اتنين columns

← لو نفتكر الصورة الأولى بدل عمل Factorization لـ 3 matrices و إشارات الـ image ... الـ vectors ~~matrix~~ maximum number of independent vectors

حيث تكون الأعمدة المعلومات المهمة هي الـ independent لقيناهم 390 مش فارق حاجة بسبب إني حافظت بالـ info الأهم فلو جيت قليت حاجة من 390 كده بدأت تقل معلومات كتير.

لو كانت الـ rank يساوي عدد الصفوف (Row) أو الأعمدة Full Rank Matrix

يعني Full Row Rank $^{3\times3}$, Full Column Rank $^{3\times2}$ ... rank = (column)

الداتا لاتتغير ولكن محور التغيير هو اللي بيتغير لانه basis مختلفة فقط ولكن كمكان لايتغير

Object كمتجهات الـ vector بدلالة basis مختلفة Date / /

ان هنا تغير كده الـ vector مكانه متغيرش ولكن الـ :- Changing basis

features الـ البعيد هقلل التغيير بس. هنحتاج موضوع الـ Changing Basis فـ

في الـ Dimension Reduction. فهو اللي هنستخدمه في حالة Changing Basis

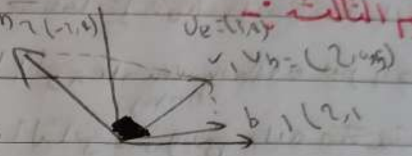
v1, v2 orthogonal وفي حالة الـ non

اطلعوا متعامدة علشان الـ linear combination

اليوم الثالث:

$v_1 = a b_1 + b b_2$ متجهات متعامدة

$v_1 = \frac{\bar{v}\cdot\bar{b}_1}{\|b_1\|^2}\bar{b}_1 + \frac{v\cdot b_2}{\|b_2\|^2}\bar{b}_2$



$v_b = 2\bar{b}_1 + 0.5\bar{b}_2$

$v_e = 2\begin{bmatrix}2\\1\end{bmatrix} + 0.5\begin{bmatrix}-2\\4\end{bmatrix} = \begin{bmatrix}3\\4\end{bmatrix}$

b basis الـ

$v_e = \underbrace{\begin{bmatrix}2 & -2\\1 & 4\end{bmatrix}}_{B} \begin{bmatrix}2\\0.5\end{bmatrix}$ E^{-1}

$v_e = b\ basis\ v_b$

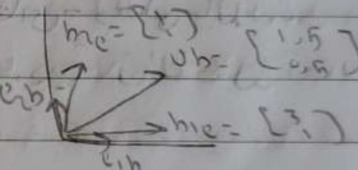
معناه ان الـ v_b = الـ basis الـ e

الـ matrix اللي فوق B E^e

(matrix هنستخدمها انها تتنقل changing للـ basis لو مكانتش متعامدة)

كده المثال ده كان عادي

طب لو مش عادي



$v_b = 1.5b_1 + 0.5b_2$

$v_e = \begin{bmatrix}3 & 1\\1 & 1\end{bmatrix} v_b$ الـ b

$v_b = [\ \ ]^{-1} v_e$

$\begin{bmatrix}1.5\\0.5\end{bmatrix} = \begin{bmatrix}3 & 1\\1 & 1\end{bmatrix}^{-1}\begin{bmatrix}5\\2\end{bmatrix}$

$B^{-1} = E$

$B = E^{-1}$ بس

ان ده عادة بتكون الـ transformation علشان المحاور orthonormal لو الـ basis orthonormal متعامدة

$\begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}$ كمان الـ orthonormal

اي لازمة الكلام ده؟

مش هتعرف تحدد الـ transformation لو مكانتش المحاور متعامدة فلو مثلا المحاور مش متعامدة لو ان المثال السابق نحدد مكان الـ b الـ ... بس اعمل الجبر

$v_e = \begin{bmatrix}3 & 1\\1 & 1\end{bmatrix} \cdot v_b$

ده بالطريقة دي الـ transformation العادي بس انا مش عايز

الناتج الموضح في الآخر بعد الـ transformation بدلالة e نقوم نرجعه تاني لـ b

$v_b = B^{-1}\begin{bmatrix}1 & 0\\0 & -1\end{bmatrix}B v_b$

اغلب الـ transformation بتتم على المحاور الأساسية بس ممكن لو حد عاوز يعمل على اي محاور مش هيقول لا بس صعب اني اعرف الحاجة المنتج $T B v_b$

$\boxed{B^{-1} T B}\, v_b$

T_b

Object على حسب التفسير اللي بعملوه. ← {orthogonalization / diagonalization} Date / /

1) في ال n-dimension حساب ال β^{-1} حسبه صعبة

2) لو عندي عاوز اعمل reflection لل vector حوليه axis بنفسه

reflection حوله

من حوليه ال e1 نوع e2

فمحتاج ال basis orthogonal هيفرق e1 و e2 البعض واحد ال axis والتاني العمودي عليه
n-1

وسيبوه يقابل e1 و e2 غير اللي عاوز ال B دة ال β^{-1} ياخدوا بتحسب بطريقة

مختلفة عن الطبيعي علشان هيبقى عملية صعبة حسابيًا فهنعمل حاجة اسمها

orthogonalization ← نحول ال non orthogonal basis إلى orthonormal basis بس واحد
n-1

منهم (العاوز reflect ال vector حوليه هو واحد منهم) والتاني عمودي عليه فكدة في

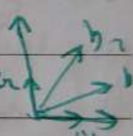
axis عرفنا + ال B هيتغير حاجة اسمها orthogonal matrix يعني $\beta^T = \beta^{-1}$ ودة

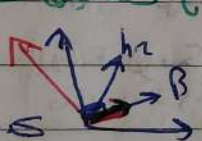
بعمل عملية غير مكلفة.

* علشان نطبق ال transformation بالطريقة الصحيحة لازم تكون ال basis متعامدة

من شرط تكون e1 و e2 الي $\begin{bmatrix}1&0\\0&1\end{bmatrix}$ أو متعامدين بس كان وخلي بالك

تروح تحسب ال $\beta^T \beta$ $\boxed{\beta_1}$ الفل ال 9 حسابيًا طريقة في الأفضل اسهل



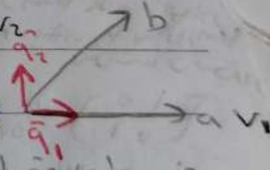


نعمل orthogonalization حولين المحاور ال B كدة عملنا orthogonalization و b

يبقى إن ال $\beta^T = \beta^{-1}$ حسابيًا بقى اسهل بكتير

→ orthogonalization :- عاوز نحولهم إلى 2 orthogonal vectors

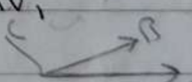
واحد منهم في اتجاه ال a وواحد عمودي على ال a هنجيبهم ازاي ؟؟؟



عن طريقة اسمها gram-schmidt ايه هي ؟؟؟ Gram Shmidt Algorithm

1) $a = \bar{v}_1$ 2) $\boxed{\bar{q}_1} = \frac{\bar{v}_1}{\|\bar{v}_1\|}$ 3) $\bar{b} = v_2 +$ Projection$(\bar{b})$ on $\bar{v}_1$

$$b = V_2 + \frac{\bar{b}\cdot\bar{v}_1}{\bar{v}_1\cdot\bar{v}_1} v_1 \rightarrow V_2 = \bar{b} - \frac{\bar{b}\cdot\bar{v}_1}{\bar{v}_1\cdot v_1}\bar{v}_1 \rightarrow \boxed{q_2} = \frac{\bar{v}_2}{\|\bar{v}_2\|}$$



وطبعًا لازم تكون المحاور دة independent .

لو 3 vectors

1) $\bar{v}_1 = a$ 2) $v_2 = b -$ project b on $\bar{v}_1$ $= b - \frac{b\cdot\bar{v}_1}{\bar{v}_1\cdot v_1}\bar{v}_1$

3) $\bar{v}_3 = c - \{$ project (c) on v_1 + project (c) on $v_2\} = c - \{\frac{c\cdot\bar{v}_1}{v_1\cdot\bar{v}_1} v_1 + \frac{c\cdot\bar{v}_2}{v_2\cdot\bar{v}_2} v_2\}$ and so on

← ودايمًا الخوارزمية دة اي الجديد ال vector دة بيضيفه ويطرح منه اي حاجة مشتركة مع

المقبلة فتخيل لو كان في vectors معتمدة على بعض يعني هيطرح كل مشترك ال سابقها

هيكون ال vector = 0 ولما نحسب ال q ← $\frac{0}{0}$ X فعلشان كدة لازم

تكون الأول إن ال vectors دة مستقلة

Object Date / /

٣ طرق تعرف من خلالهم ال vectors دول independent ولا لا ١) من مجرد النظر

لو حد عبارة عن تاني في التاني ٢) تحطهم في مصفوفة واحسب ال determinant ٣)

والأحسن في حالة ال 3×3 وبتقلل كدا ((pivot) من خلاله تشوف ال rank بتاعهم.

طب بنعمل ليه اصلاً ال orthogonalization ما $v_1 = \{1, 1, 1\}$, $v_2 = \{2, 0, 1\}$, $v_3 = \{\dots\}$

وعاوز نعمل orthogonal vectors لـ (v_1, v_2, v_3) → بالنسبة لـ v → $v_3 \{2, 3, 5\}$

في ال اعمل orthonormalization وهات $e_1 = \{\frac{v}{\dots}\}$ e_2 e_3

$$\begin{bmatrix} \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{6}} \\ \frac{1}{\sqrt{3}} & \frac{-1}{\sqrt{2}} & \frac{1}{\sqrt{6}} \\ \frac{1}{\sqrt{3}} & 0 & \frac{-2}{\sqrt{6}} \end{bmatrix} \rightarrow \underset{\text{matrics}}{E} \rightarrow E \text{ basis with respect to V basis}$$

فلو عاوز تعمل transformation لـ r_v

$$u_e = [E]^{-1} r_v$$

بعد ما تخلص خالص هو حايبدا بتحويه بدلة ال non orthogonal تحت بعد ما تكون جبت ال transformation

$E T E^{-1} r_v$

$E \to$ orthonormal and | 1=1 → orthonormal $\boxed{T_v}$

$E^{-1} = E^T$

$\boxed{E T E^T r_v}$ r_v

ده بقا ال transformation المتعمل لو المحاور $\leftarrow T_v$ r_v

ما تهولش بدلالة ال non orthogonal يبدو ال non ortho الفكرة الأولى

ال E ده بيسموها Q يبدو ⇐ الطريقة التانية الممكن اجيب منها ال E matrix

ده مبادرة السها Q يبدو ← matrix factorization من طرق اتوائها

لو عندي square matrix V ← $\begin{bmatrix} | & | & | \\ v_1 & v_2 & v_3 \\ | & | & | \end{bmatrix}$ نقدر نقسمها لـ 2 matrices $V = \boxed{Q}\,\boxed{R}$

في الحالة دي shape upper triangular matrix

$R = Q^{-1} V = Q^T V$

np.linalg.qr($\hat{V}$) Q, R

decomposition

حل لو ابتدت بتجيب ال e طلع ماله component سالبة عادي طالما نفس ال span فرق

في ال span طلع نفس الـ vector بس مغير ال direction عادي بس بيكون مغير ال span

$\begin{bmatrix} 5 & 2 \\ 2 & ? \end{bmatrix}$

عندي Tm بتعملو لل Vector كلو لما بتغيرو كلو في بعض Rule

Characteristic → eigen vector اسمها characteristic لل Vector ال من ال Vectors Rule
transformation matrix

eigenvalues 8

Eigenvectors and Eigenvalues :- ال eigenvectors مرتبطة بال transformation المعمولة

ال matrix اللي هي ← * transformation لما بتعمل ال transformation لل Vector كل ال

لأن ممكن يبقى بالسالب direction

Vectors (المتغيرين) ال (span) direction بتاعها بنسميها (eigenvectors)

(ال Vector الخطواتها اللي بنسميها eigenvectors) ومقدار ال scale اللي حصل بيقي ال eigenvalue

ممكن يبقى ال eigenvalue بالسالب ومعناه إن ال Vector اتعكس بالسالب

← ال eigen vector ممكن يكون أي vector في ال span دا كله.

← عدد ال independent eigenvectors في ال 2d ← 2 لأن ال ... كلو ...

فال maximum 2 ممكن يبقى واحد بس على طول ... eigenvectors dependent

* لو ال X eigenvector لل matrix A يبقى ... A

$Ax = \lambda x$

transformation لل X Vector الناتجة ...

لما عملت ال transformation لل eigenvector ... ال vector مضروب في scale (λ)

⇐ يجب إن ال eigenvalues and eigenvectors باي ... numpy

np.linalg.eig(v)

1) Diagonalization

شرطها إن ال eigen vectors يكونوا independent

$$\left(\begin{bmatrix} 5 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 2 \end{bmatrix}\right)^{1000} = \begin{bmatrix} 5^{1000} & 0 & 0 \\ 0 & 3^{1000} & 0 \\ 0 & 0 & 2^{1000} \end{bmatrix}$$

أما لو ال matrix دي هنضربها في نفسها 1000 مرة عملية صعبة جدا

... Eigen Decomposition ← لو ...

T matrix ... diagonizable matrix يعني لو 2x2 يبقى ليها 2 independent eigen vectors

ال ... T ... eigen Decomposition

$$C D C^{-1} \quad \underbrace{\begin{bmatrix} | & | \\ v_1 & v_2 \\ | & | \end{bmatrix}}_{\text{eigen vectors with respect to original basis}} \underbrace{\begin{bmatrix} \lambda_1 & \\ & \lambda_2 \end{bmatrix}}_{\text{Transformation matrix in eigen space}} \underbrace{\begin{bmatrix} | & | \\ v_1 & v_2 \\ | & | \end{bmatrix}^{-1}}_{\text{original basis with respect to eigen basis}}$$

eigenvectors of matrix T

diagonal eigen values

Decomposition

$$A^{100} = \underset{\text{eigen vector}}{C}\ \underset{\text{eigen values diagonal}}{D^{100}}\ \underset{\text{eigen vectors}}{C^{-1}}$$

matrix_power in numpy

٣) معلومة جانبية ← كل ما ال features تزيد (الأبعاد) كل النقط بتبقى تقريبًا بتبقى على نفس البعد من بعض ودة بيخلي حاجات زي ال KNN و ال Clustering (اللي بيعتمدوا على المسافات) متعرفش تحدد مين شبه مين لاء.

Object Date / /

ال rank بيخليني أختار عدد ال Principle Components

← في حاجة تبقى أحسن منه إني أعمل $T = LD\boxed{L^{-1}}$ فلو كانت ال L ← orthonormal

فهيبقى ال L^{-1} هو ال L^T فهيبقى أحسن أكيد ودة بالنسبة ال covariance matrix)

ومهم في ال Dimensional reduction بيتغير كتير:

→ Dimensionality reduction :- $\overbrace{x_1, x_2, \dots, x_n}^{n}$

فمشكلتها ← ١) ال Learning Algorithm بيبطأ $m*n$ لو كان ال dimension كتير

← ٢) ال Visualization لو ال dimension كتير فصعب أعرف أرسم

فنختار لو حولتها $x_1 \to x_n$ وحولتهم ل 2 features بس z_1 | z_2 | ودول

يحتووا على 95% من ال information في الداتا الأصلية فدة أكيد أحسن بكتير

مشكلة ال dimension دة بيسموها ← (curse of dimensionality) اللعنة و ال accuracy

ملوش علاقة بالموضوع الهدف الأساسي السببين اللي قلناهم فوقه ممكن بالصدفة

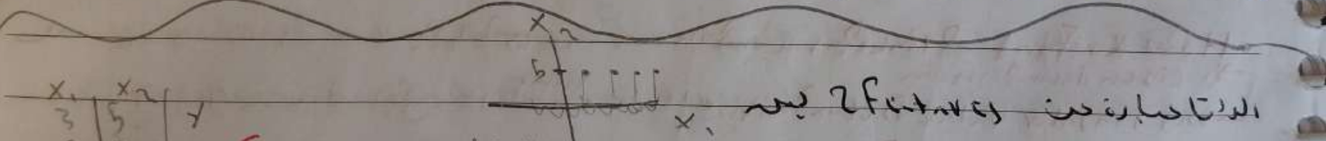
وأحيانًا أكون شلت noise بيطلوش دقة. فالقاعدة كالآتي:

لو الداتا اللي عندي أعملها training و analysis من غير ما أعمل dimension reduction

فدة هيكون أحسن طبعًا لو لقيت ال learning algorithms بطيء ومش عارف أعمل

analysis فهستخدم ال dimension reduction. في العادة زي ما قلنا كدة بنحاول نجيب

ال most information (Highest variance)



x_1	x_2	y
3	5	
4	5	
5	5	
6	5	

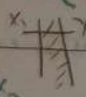
الداتا عبارة عن 2 features بس

x_1 بيتغير و x_2 ثابتة جدًا x_2 ليها لازمة ولا أشيلها ← فممكن

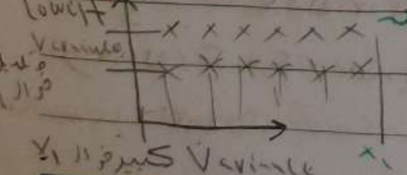
أستغنى عنها لأنها معهاش أي نوع من أنواع التغير (معهاش أي Variance)

فلو عندي ال y لا يعتمد على x_2 في حاجة أصلًا y بيتغير بتغير ال x_1 فلو شلت

x_2، كدة عملت projection على x_1 فالتالي شلت ال component بتاع x_2



كدة عملت dimension reduction من 2D ← 1D



لو عملت projection عن x_1 حذفت ال component ال x_1 ونفس

الكلام على x_2 لو شلنا ال variance بتاع x_1 بيمثل 98% من

قيم البيانات اللي فلو شلت ال x_2 كدة خسرت ١٠٪ بس

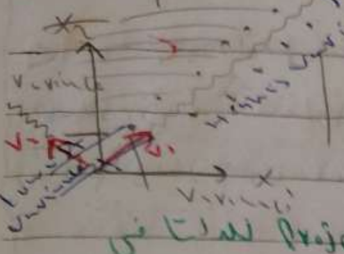
من البيانات. و ال projection هو ال dot product

في ال n dimension ال يكون قريب من ال component والباقي
فيهم صغيرة يعني ~~صفر~~ صفر غير
zero →

لما بنوصل لل rank بتخسر ترجاجه من الداتا
لو قللت عن ال rank ده بتبدأ أتكسر بيانات
معاكد

Object

Date / /



* فلو عملت Projection على x₁ هلاقي 90% من الداتا في ال x₁
وفي ال variance على x₂ ← 10% ساعتها ممكن أعمل dimension reduction
طيب لو عملت ال axis مايل بزاوية 45° ال

فلو v₁, v₂ دول (unit vectors) وعملت Projection للداتا في
اتجاه v₁ وفي اتجاه v₂ هنلاقي الداتا بدل x₁ | x₂ ← z₁ | z₂

دور ساعتها اسمهم ال components z₁ و z₂ وكل كمية الداتا متغيرتش
لأن كل اللي عملته هو changing basis يعني الداتا محتوية بدلالة المحاور الجديدة
← فلو الداتا فيها correlation بين x₁, x₂ فكرة بيوركز ال vector ال
orthonormal وعمل projection للداتا في اتجاه ال highest variance وهي ده المجدها
واستغنى عن ال component التانية (ده حوال dimension reduction)
← بيكون عندك عدد ال dimension = dimensional ال الداتا بس اختار ليها اتجاهات ال vectors
وده يعني لو عملت projection على ال 3 دول يبقى أول واحد فيهم ال highest variance
فيكون ال projection على أول vector بيدي أكبر component على v₂ الأهمية على v₃ البعيدة

V x₁ x₂ x₃ x₄
5 6 7 8

z₁ | z₂ z₁ | z₂

How to find Principal Component axes ??!!   هو x₁ → xₙ

1) eigen decomposition
2) singular value decomposition

هنا خ n ال dimension جاي كل واحد n dimension
ودول الطريقتين المجيبين من خلالهم P(a×b)
معرفة ال x وال y هل العلاقة دي طردية وبحدة او عكسية → covariance →
هل هي علاقة خطية ولا لا ولكن لا يستطيع تحديد قوة العلاقة دي
ال covariance بين المتغير والعلاقة، بس ال variance لل feature دي نفسها
← لا يحدد من الرقم قوة العلاقة علشان بيتأثر بال scale فمحتاج normalization بين -1, 1
وتسميه ← correlation

$$\text{Correlation} = \frac{\text{Covariance}(x,y)}{\sigma_x \cdot \sigma_y}$$

لو عندي أكتر من 2 feature بنحط كل المعلومات دي في مصفوفة اسمها
Covariance matrix

Object Date / /

مصفوفة الـ covariance matrix بتكون مصفوفة مربعة و symmetric

ولو عملنالها normalization و قسمناها على حاصل ضرب الـ standard deviations هينتج الـ correlation matrix

وتأكيد القطر الرئيسي هيكون عبارة عن 1.

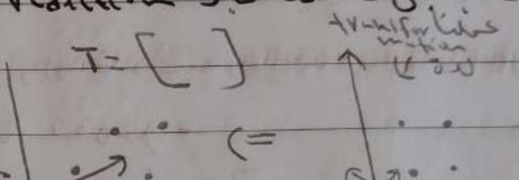
الـ symmetric تتميز بإنها Full orthonormal eigenvectors يعني لو جبت الـ

eigenvectors للـ covariance matrix هطلعوا orthonormal vectors معناه إن الـ inverse لـ

الـ eigenvectors هو الـ transpose

٣) إن الـ covariance matrix بيسموها semidefinite matrix ← (≥0) eigenvalues ← عدد الـ redundant column

طب إي علاقة بالـ dimension reduction.



هطلع الـ v1 , v2 eigenvectors ولو نشيل الـ direction بتوع eigen vectors

و $\lambda_1 > \lambda_2$ لأن الـ variance عالية أكتر وأكبر

هتلاقي الـ v1 رايحة في اتجاه الـ maximum variance و v2

في اتجاه الـ lowest variance . الـ T دة هو الـ covariance matrix diagonal مصفوفة الـ $Co = \underset{\sim}{V} \underset{\sim}{D}$

الـ v1 دة هو الـ projection direction للبيانات عليه و نهمل الـ v2 و بكدة عرفنا

الـ PC1 PC2, PCn, ... components

الـ covariance matrix من المتوقع بشكل الـ transformation اللي حصل من الداتا الخام

تغير كدة يعني تتغير الداتا كأنها عملنالها transformation و حطيناها في transformation matrix ;

الـ matrix دة هي المصفوفة قوية بالـ covariance matrix قدرة من المتوقع الـ transformation دة تعبيرها

خلالها نجيب الـ eigenvectors اللي أكبر eigenvalues يبقى الـ PC1 و ..... $Co = Q \Lambda Q^T$

لو $\mu = 0$ ← الـ $Co = \frac{X^T X}{n-1}$

مشكلة الـ eigen Decomposition إنه مبيشتغلش (SVD) Singular Value Decomposition

غير لما المصفوفة تكون مربعة و لازم علشان نجيب الـ PC1 و PC2, ... لازم نعمل

Eigen Decomposition للـ covariance matrix (X) يعني نحسب الـ λ و الـ eigenvectors المقابلة لها وده

مشكلة كبيرة لو الداتا الـ features بتاعتها كثيرة إني أحسب الـ covariance matrix وأعملها

decomposition عملية صعبة حسابيًا فبنروح لـ SVD اللي بتعمل decomposition للـ non square matrix

ومباشرة أجيب الـ PC1 , PC2 , ...

في علاقة بين الـ SVD و الـ eigen decomposition هي بالـ.

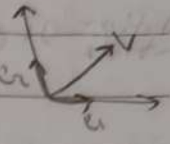
سؤال بحثي حد في حالات بيكون فيها Eigen Decomposition أحسن من الـ SVD ؟؟؟؟

Object .... Date / /

لو الداتا اللي عندي كمان مصفوفة غير مربعة مش هنعرف نعملها Eigen Decomposition

SVD → ماهو إلا انه بيـ decompose الـ vectors إلى الـ orthogonal axes

$\vec{V} = \boxed{5}\boxed{\hat{e}_1} + \boxed{3}\boxed{\hat{e}_2}$ → عملت decomposition للـ vector لـ original components بتاعه



Singular values for this vector

Singular vectors

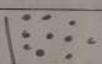
عشان كده معنى inverse لو عملتهم projection على الـ orthogonal axes

دور البعد علية إني أجيب الـ PC1 , PC2 اللي هنعمل عليهم projection

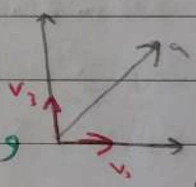
الـ SVD ← بتخليني مباشرة أجيب الـ V1 والـ V2 اللي كنا بنجيبهم زمان من خلال الـ covariance matrix . الـ singular values matrix بتاع على الـ axes (V_1, V_2) دول الـ PC هما الـ singular vectors

فلو جبتهم بيبقى جبت الـ Principal Components . طب إزاي بقا أجيب الـ singular vectors دول من غير ما استخدم الـ covariance matrix اللي احنا عاوزين نفهمه .

$a = S_{a1} \bar{V}_1 + S_{a2} \bar{V}_2$



بيكون عندي نقط كتيرة في الداتا كل واحدة من دول عبارة عن vector



وحلل كل نقطة لمجموعة الـ orthogonal ولو فرضت الأفق الـ axes orthonormal

اللي بيديني الـ highest variance كده جبت الـ PCs اللي كنا بنجيبهم زمان طبعا

مش لازم يكونوا في اتجاه لا لا المهم عندي إن يكونوا orthonormal لبعض

الـ projection على الـ axes بتاعتي

$A \cdot V = S$

$$\begin{bmatrix} a_x & a_y \\ \vdots & \vdots \\ h_x & h_y \end{bmatrix} \cdot \begin{bmatrix} V_{1x} & V_{2x} \\ V_{1y} & V_{2y} \end{bmatrix} = \begin{bmatrix} S_{a1} & S_{a2} \\ \vdots & \vdots \\ S_{h1} & S_{h2} \end{bmatrix}$$

الداتا

الـ vectors المتعامدة بتاعتي

الإسقاطات من الـ vector لقيم إتجاهها

كل الإسقاطات في اتجاه V_1

$$\underset{n \times 2}{\begin{pmatrix} a_x & a_y \\ h_x & h_y \\ \vdots & \vdots \end{pmatrix}} \underset{2 \times 2}{\begin{pmatrix} V_{1x} & V_{2x} \\ V_{1y} & V_{2y} \end{pmatrix}} = \underset{n \times 2}{\begin{pmatrix} S_{a1} & S_{a2} \\ S_{h1} & S_{h2} \\ \vdots & \vdots \end{pmatrix}}$$

unit vector

→ $A \cdot V = S \rightarrow A = S V^{-1} = A = S V^T$

orthonormal → orthogonal

projection

فهنا كل الـ vectors الـ V الموجودة عاوز أعمل أسوي منهم الـ highest variance ← الـ V (vectors)

إزاي بقا !!!

تخيل لو الداتا منتشرة [illegible] وحبيت تطبق عليها Transformation (Covariance) لتلاقيها
حدث شكل بيضاوي كده [illegible] أكثر vector بعيد عن الداتا ده تحوار الـ vector الرسمورع وهو

Object تحواة الـ eigenvector علشان اتجاهه متغيرش Date / /

Projection on V_1

$S = \begin{pmatrix} S_{a1} & S_{b1} \\ S_{a2} & S_{b2} \end{pmatrix}$ on V_2

لو حسبنا الـ projection في اتجاه V_1 و اتجاه V_2 لكل الـ Projection

ده projection اتجاه V_1 > projection اتجاه V_2 يبقى V_1 هو الـ PC1 و التاني الجديد وكده

فالأعلى projection لـ vector يعتبر معناه ان الاتجاه بتاعه هو الفيه كمية اكبر من البيانات

ـه لأطول فرد الـ variation الإشعاع بتاعه هو الـ PC1 وده بتحسب عن طريق:

Singular Values
$$\begin{cases} \sigma_1 = norm = \sqrt{(S_{a1})^2 + (S_{a2})^2 + \cdots} \\ \sigma_2 = \cdots\cdots \quad 2 \quad 2 \cdots \end{cases}$$

$$S \begin{cases} \rightarrow V \\ \rightarrow \Sigma \end{cases}$$

$A = \textcircled{S} V^T = U \Sigma V^T$

eigen vector
transformation بنفس

$U \leftarrow \frac{S_{a1}}{\sigma_1}$

هنعمل normalization لـ S على قيمة الـ σ بتاعته يتحول إلى U، انحول الـ projection vectors

وعلشان اكون عملتش حاجة تضرب في σ

اقدر اقول اني هاخد الـ V اللي انا على σ

$$S = \underbrace{\begin{pmatrix} \frac{S_{a1}}{\sigma_1} & \frac{S_{a2}}{\sigma_2} \\ \frac{S_{b1}}{\sigma_1} & \frac{S_{b2}}{\sigma_2} \end{pmatrix}}_{U} \underbrace{\begin{pmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{pmatrix}}_{\text{Singular Values}} \rightarrow \sigma$$

Projection
normalized → unit vector

لو جبت الـ Covariance matrix لما بجيبلها الـ eigen Decomposition الـ eigen vectors بتوعها هي الـ Principle Components

علاقة الـ eigen Decomposition بالـ SVD !!!!!

$\boxed{A} = U \Sigma V^T$

$U^T = U^{-1}$

$$\rightarrow C_0 = \frac{A^T \cdot A}{n-1} = \frac{(U\Sigma V^T)^T \cdot (U \Sigma V^T)}{n-1} = \frac{V \Sigma^T \cancel{U^T} \cancel{U} \Sigma V^T}{n-1} =$$

$$\boxed{V} \frac{\Sigma^T \Sigma}{n-1} \boxed{V^T} \qquad \boxed{\lambda = \frac{\Sigma^T \Sigma}{n-1}} = \boxed{\lambda = \frac{\Sigma^2}{n-1}}$$