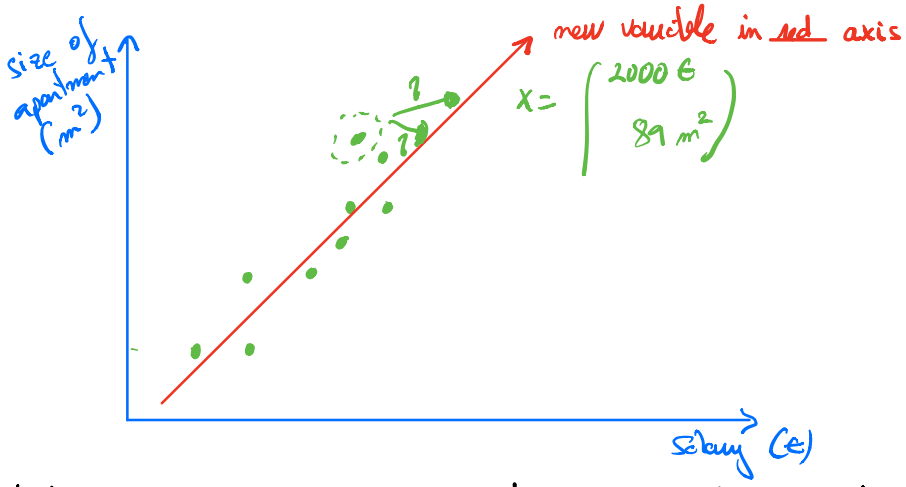
Introduction

What does **dimensionality reduction** means?

> Reduce data from N-dimension space to a lower space
>
> ⇓
>
> features



Which is the motivation to apply dimensionality reduction?

(1) Compression of data → e.g. for video
(2) Reduce noise → e.g. for images and video
(3) Reduce the memory to process data
(4) Reduce the storage
(5) Accelerate analytics → models are executed faster
(6) Recommendation engines
(7) Data visualization of high dimensional dataset

Example of data visualization

Data Visualization

Country	x_1 GDP (trillions of US$)	x_2 Per capita GDP (thousands of intl. $)	x_3 Human Develop-ment Index	x_4 Life expectancy	x_5 Poverty Index (Gini as percentage)	x_6 Mean household income (thousands of US$)	...
Canada	1.577	39.17	0.908	80.7	32.6	67.293	...
China	5.878	7.54	0.687	73	46.9	10.22	...
India	1.632	3.41	0.547	64.7	36.8	0.735	...
Russia	1.48	19.84	0.755	65.5	39.9	0.72	...
Singapore	0.223	56.69	0.866	80	42.5	67.1	...
USA	14.527	46.86	0.91	78.3	40.8	84.3	...
...	...	...	...	...	...	...	...

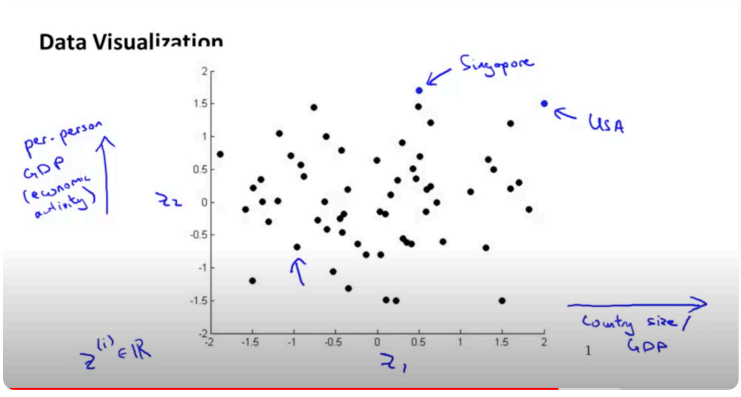
Dimension reduction

Data Visualization

$z^{(i)} \in \mathbb{R}^2$

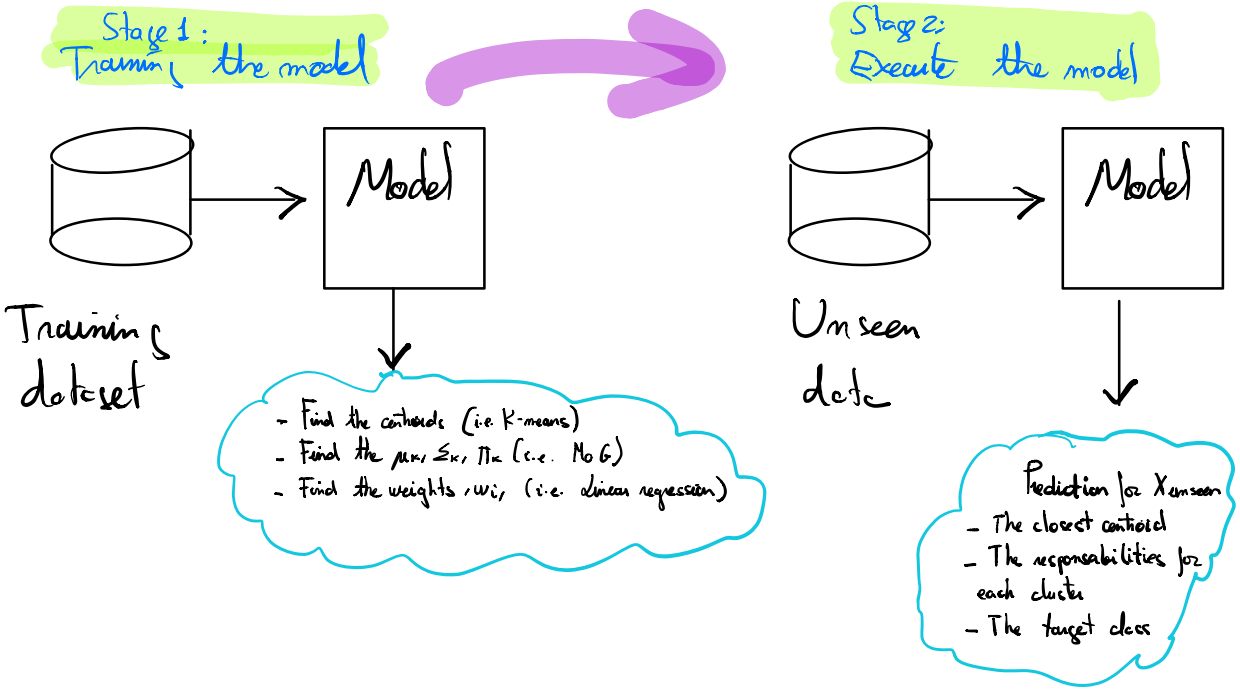
Country	z_1	z_2
Canada	1.6	1.2
China	1.7	0.3
India	1.6	0.2
Russia	1.4	0.5
Singapore	0.5	1.7
USA	2	1.5
...	...	...

Data Visualization

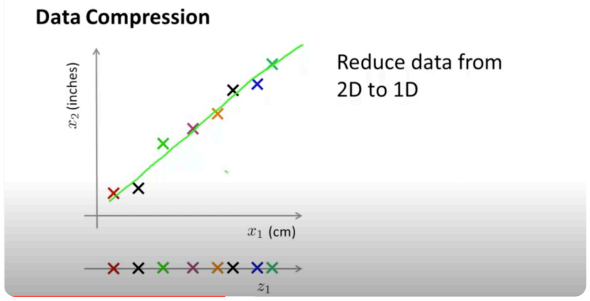


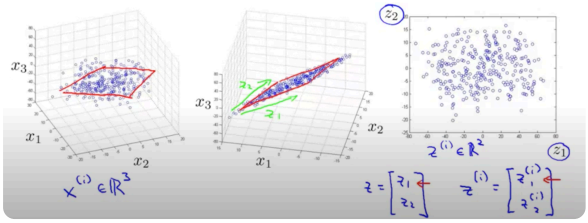
It's easier to understand the data!!

Let's come back to our goal in Machine Learning....



⇒ Therefore, in dimensionality reduction, our goal in the training stage is to find a new dimensional space. In other words... we need to find a mathematical expression to move from N dimensional space to an optimal new P dimensional space where P < N (e.g. 1000 → 100D)

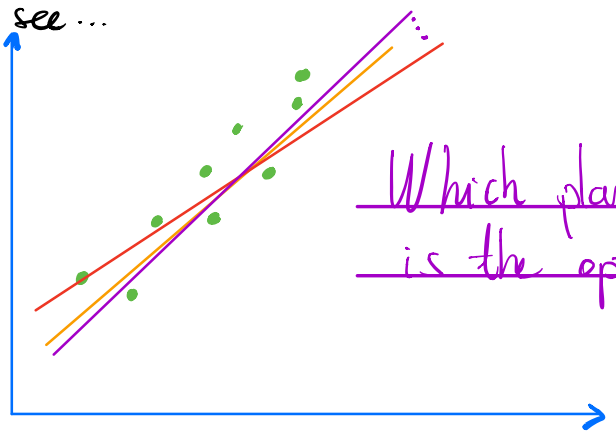




Data Compression
Reduce data from 3D to 2D

How to find the optimal new space?

Let's see...



Which plane is the optimal one?

$x^{(1)} \in \mathbb{R}^2 \longrightarrow z^{(1)} \in \mathbb{R}^1$

$x^{(2)} \in \mathbb{R}^2 \longrightarrow z^{(2)} \in \mathbb{R}^1$

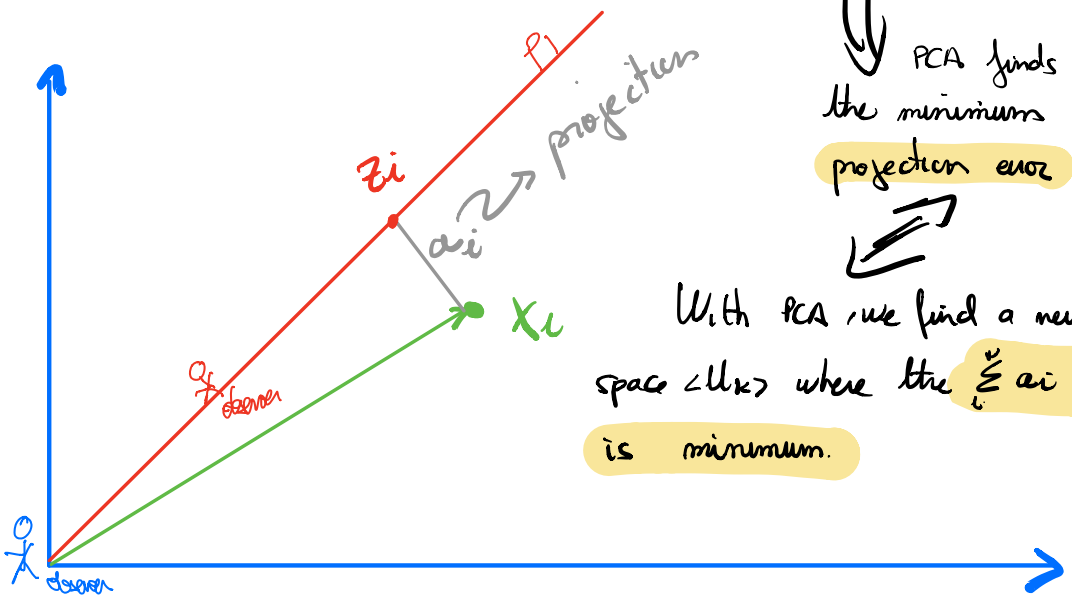
$\dots$

$x^{(m)} \in \mathbb{R}^2 \longrightarrow z^{(m)} \in \mathbb{R}^1$

Solution = Principal Component Analysis looks for a new hyperplane (i.e. dimensional space) where the distance of the training dataset's points and their projections are minimal

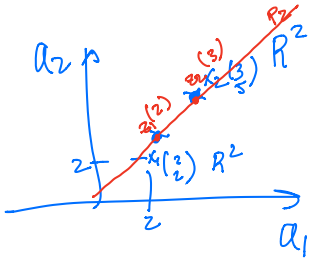
⇓ PCA finds the minimum projection error

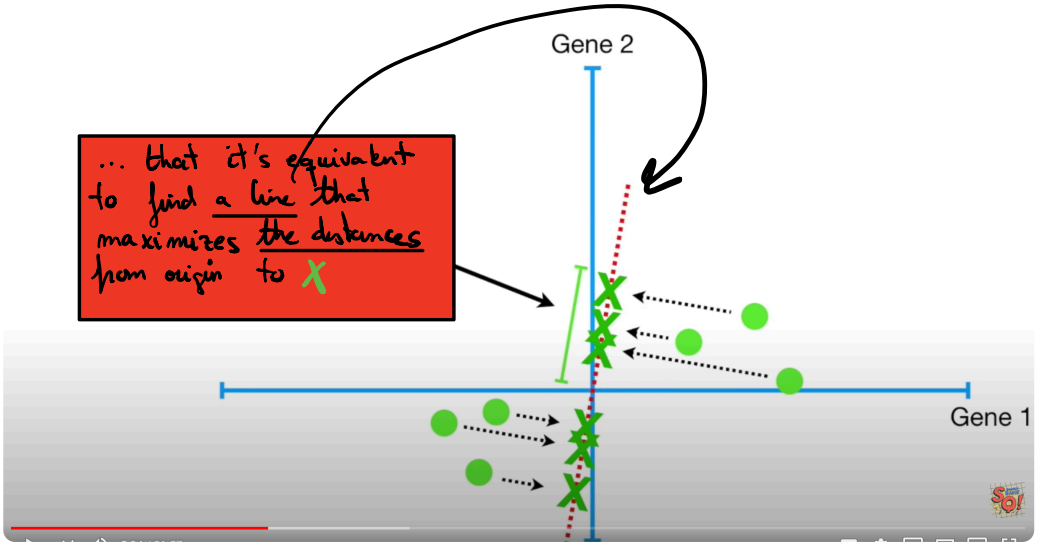
⇓ With PCA, we find a new space $\langle u_k \rangle$ where the $\sum_i a_i$ is minimum.



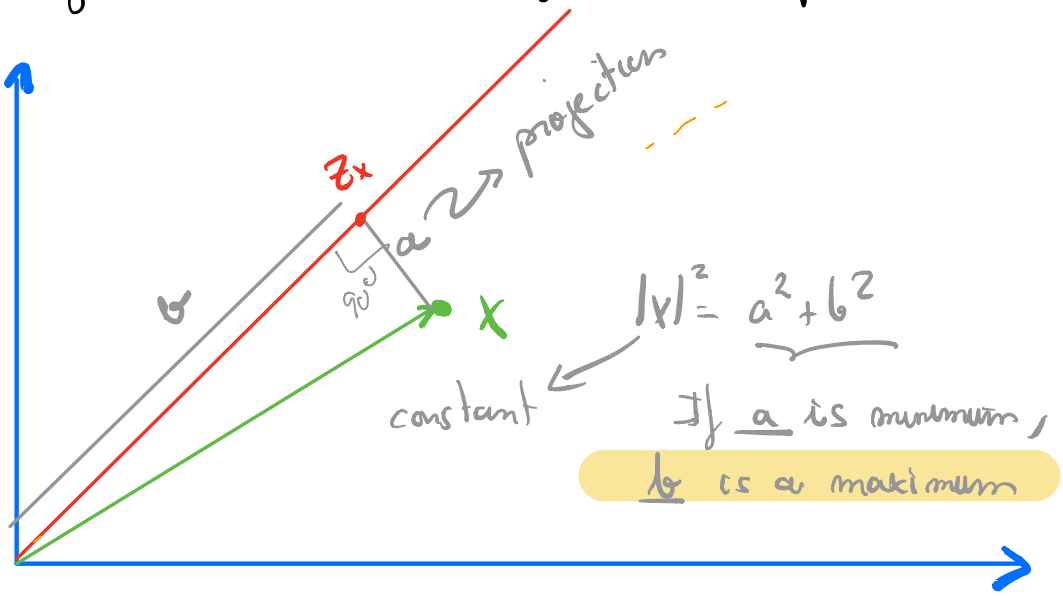
So, next question is... How to find a new space where the total projection error is a minimum ??<u>

Let's come back to our plot....





If we do zoom in just one datapoint...

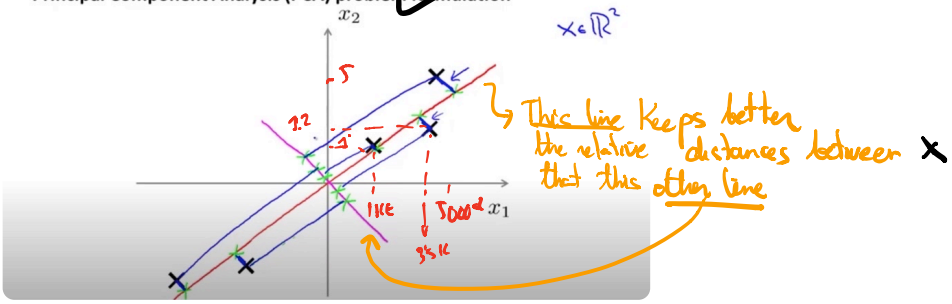


Therefore, the goal in PCA is to find a new space $\langle \vec{u} \rangle$ that captures the maximum variance (i.e. b is maximum)

maximum relative distances

minimum projection

Principal Component Analysis (PCA) problem formulation



In summary, PCA calculates the directions $\langle \vec{u} \rangle$ of greatest variance. As eigenvectors are in the directions of maximum variance, eigenvectors are in the bases of PCA

Training Stage: How to calculate PCA of a dataset X?

Imagine X as ....

	# cars	salary	size home	# members
Family 1	2	3000	180	5
Family 2	1	2850	95	3
...	...	...	...	...
Family N	3	4250	120	4

	X_1	X_2	X_3	X_4
$x^{(1)}$	2	3000	180	5
$x^{(2)}$	1	2850	95	3
...	...	...	...	...
$x^{(n)}$	3	4250	120	4

Stage 1 → For each feature, we calculate the mean:

$$\mu_j = \frac{1}{N} \sum_{i=1}^{N} x_j^{(i)}$$ → For feature or variable (D dimensions)

↳ If features have different scale, it makes sense to rescale the values to have comparable range of values

Stage 2 → Remove the mean of each feature to each datapoint ⇔ It's equivalent to center the datapoints in the origin

$$x \longrightarrow \bar{x} = x - \mu$$

Stage 3 → We calculate the "covariance matrix" of $\vec{X}$, i.e. Σ

Stage 4 → Find eigenvectors and eigenvalues of Σ using SVD:

$$\Sigma = U^{-1} \cdot D \cdot U$$

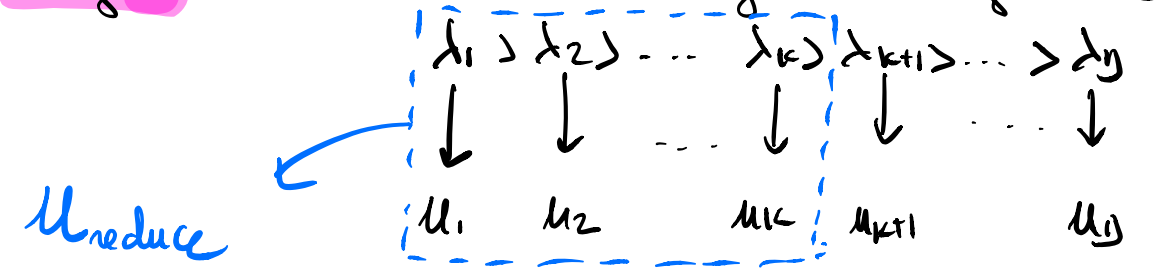
where U is the matrix of eigenvectors of Σ, i.e. $U = \begin{bmatrix} | & | & & | \\ u_1 & u_2 & \dots & u_D \\ | & | & & | \end{bmatrix}$

where D is the matrix of eigenvalues of Σ, i.e. $D = \begin{bmatrix} \lambda_1 & 0 & \dots & 0 \\ 0 & \lambda_2 & \dots & 0 \\ 0 & \dots & & \lambda_D \end{bmatrix}$

Stage 5 → We order eigenvectors based on their eigenvalues: $\lambda_1 > \lambda_2 > \dots > \lambda_D$

↳ greatest eigenvalues, greatest variance

Stage 6 → We select the first k eigenvectors



Stage 7 → We are ready to project datapoints in the new U_{reduce} space as follows:

$$\underline{x} = U_{reduce}^{-1} \cdot \bar{x}$$

$$U_{reduce}^{-1} = U_{reduce}^{T}$$ because U_{reduce} is orthonormal

We should take $\leq$ eigenvectors which explain the 80%-95% of total variance

Stage 8 → Come back to original coordinate bases:

$$\bar{x}_{reprojected} = U_{reduce} \cdot \underline{x}$$

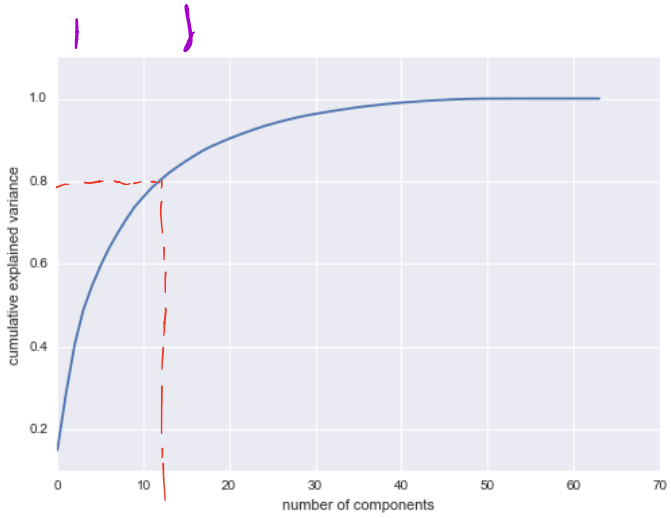
$$x_{reprojected} = \bar{x}_{reprojected} + \mu$$

To evaluate the performance, we calculate the

$$\text{Proportion of variance} = \frac{\sum_{i=1}^{K} \lambda_i}{\sum_{i=1}^{D} \lambda_i}$$

K = selected K components from PCA

D = dimension of original dataset



Summary

Review / Definition of PCA

→ systematized way to transform input features into principal components

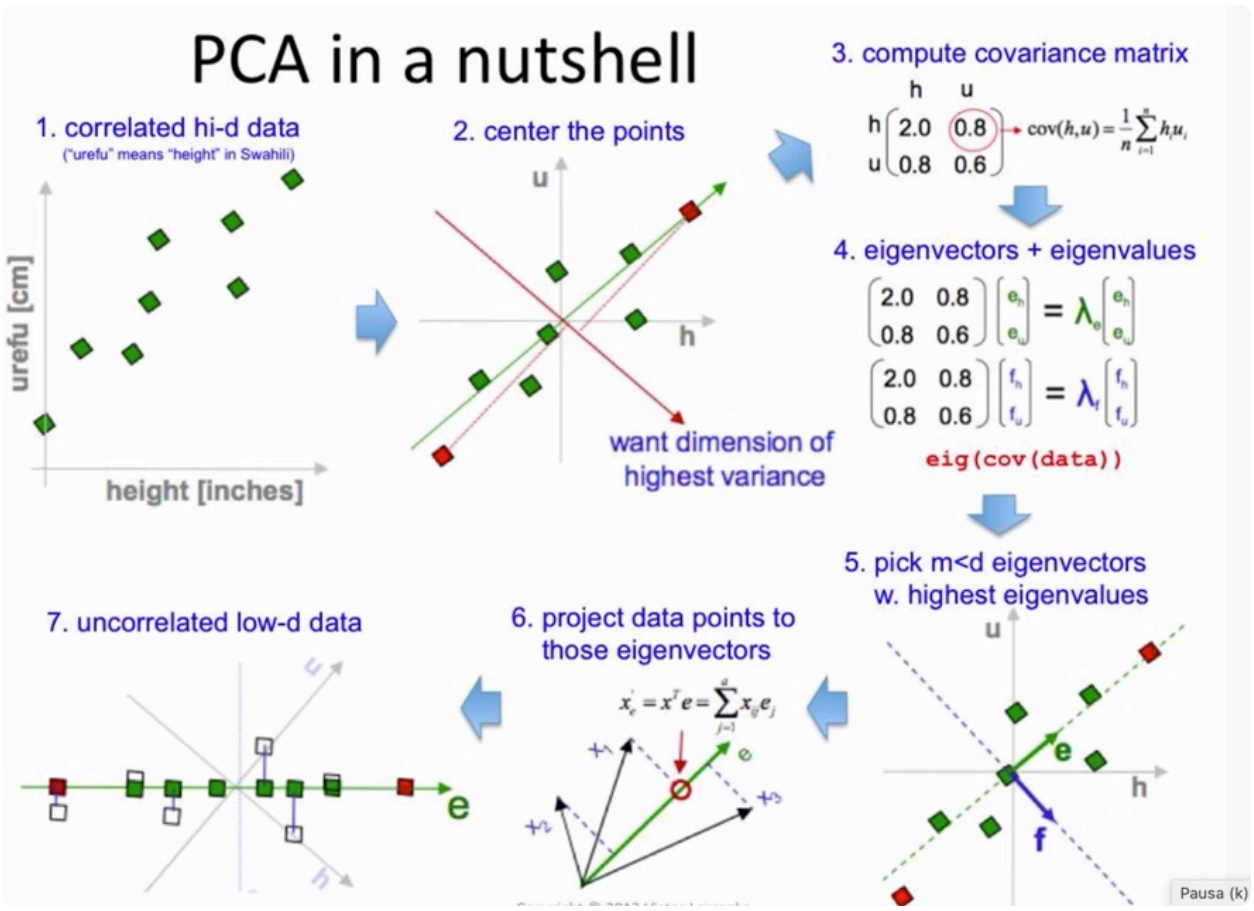
→ use principal components as new features

→ PCs are directions in data that maximize variance (minimize information loss) when you project / compress down onto them

→ more variance of data along a PC, higher that PC is ranked

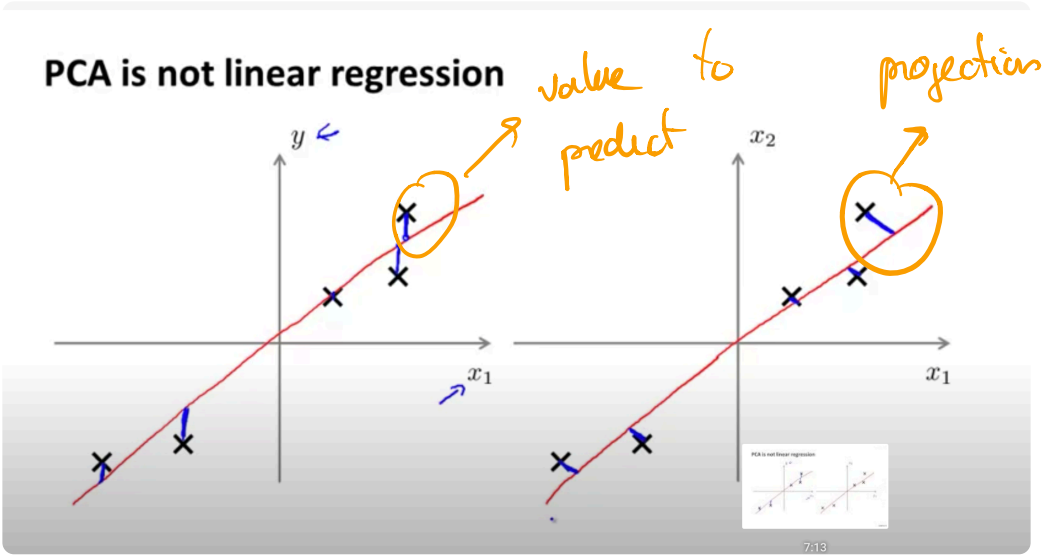
→ most variance / most information → first PC
second-most variance (without overlapping w/ first PC) → second PC

→ max no. of PCs = no. of input features



Pausa (k)

Please, no confuse PCA with linear regression



An interpretation in images: