Lecture 7: Principal Component Analysis

Q: How do we find salient directions in data?

$x_1, \dots, x_n \in \mathbb{R}^d$, d large.

Q: How do we find the "best" low dimensional representation of data?

Many applications:

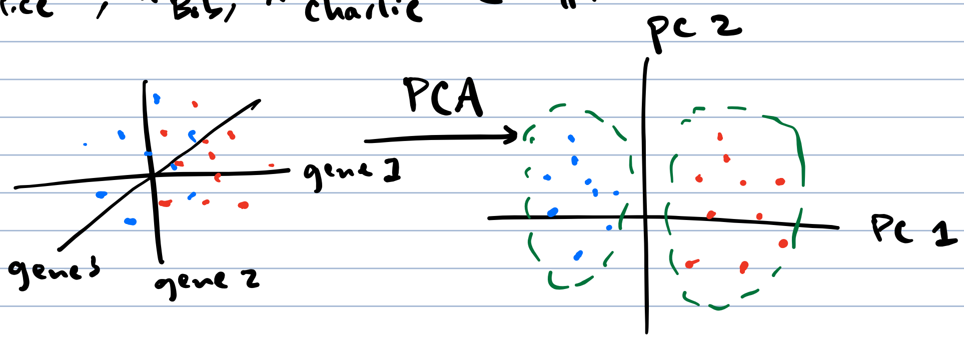
- data visualization
- discovery
- data clustering.
- ...

e.x. genetics data

	gene 1	gene 2	....		gene d
Alice	0	0	1	0 ...	0
Bob	1	0	0	1 ...	0
Charlie	0	0	1	0 ...	0
⋮					

1 if Bob has a mutation at position 1.

$x_{Alice}, x_{Bob}, x_{Charlie} \in \mathbb{R}^d$



Toy example:

4 people rate foods from 1-10.

	kale	taco bell	sashimi	pop tarts
Alice	10	1	2	7
Bob	7	2	1	10
Carol	2	9	7	3
Dave	3	6	10	2

Q: How do we visualize this data?

step 1: center the data

$$\mu = (5.5, 4.5, 5, 5.5)$$

step 2: find 2 good directions v_1, v_2 for the data s.t.

$$x - \mu \approx a_1 v_1 + a_2 v_2 \quad \forall\, x \in \{\text{Alice}, \text{Bob}, \text{Carol}, \text{Dave}\}.$$

turns out, if you take

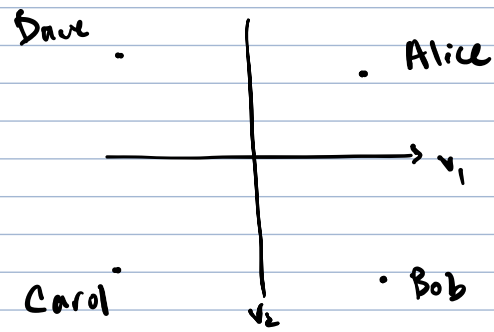
$$v_1 = (3, -3, -3, 3)$$
$$v_2 = (1, -1, 1, -1),$$

then

$$\text{Alice} - \mu \approx v_1 + v_2 \rightarrow (1, 1)$$
$$\text{Bob} - \mu \approx v_1 - v_2 \rightarrow (1, -1)$$
$$\text{Carol} - \mu \approx -v_1 - v_2 \rightarrow (-1, -1)$$
$$\text{Dave} - \mu \approx -v_1 + v_2 \rightarrow (-1, 1)$$

e.g. $v_1 + v_2 = (4, -4, -2, -2)$

$\text{Alice} - \mu = (4.5, -3.5, -3, -1.5)$ pretty close!



vegetarian-ness

big v_1 → like kale, pop-tarts, dislike TB + sashimi

big v_2 → like kale, sushi, dislike TB + pop-tarts

healthiness

we can use this to infer more properties of their food prefs!

Principal Component Analysis

Given $X_1, \cdots, X_n \in \mathbb{R}^d$

Typically, de-mean them. Let $\mu = \frac{1}{n}\sum_{i=1}^{n} X_i$,
set $X_i' = X_i - \mu$, so that new mean is $(0, \cdots, 0)$.

So to slightly simplify notation, let's just work with de-meaned data, i.e. let's assume $\mu = 0$.

Goal: Find a subspace $V \subseteq \mathbb{R}^d$, $\dim(V) = k$

so that $X_i \approx \text{proj}_V(X_i)$.

parameter.

Want $k << d$ (often constant).

More concretely:

$V = \text{span}\{v_1, \cdots, v_k\}$. We can choose $v_1, \cdots, v_k$ orthonormal

$$\langle v_i, v_j \rangle = \begin{cases} 0 & \text{if } i \neq j \\ 1 & \text{if } i = j \end{cases} \leftarrow \|v_i\|_2^2 = 1.$$

PCA objective:

$$\underset{\substack{v_1, \cdots, v_k \in \mathbb{R}^d \\ \text{orthonormal}}}{\text{argmax}} \sum_{i=1}^{n} \sum_{j=1}^{k} \langle X_i, v_j \rangle^2 \leftarrow \|\text{proj}_V(X_i)\|_2^2$$

orthonormal basis.

Interpretation: For $V = \text{span}(\{v_1, \cdots, v_k\})$, the projection of $X \in \mathbb{R}^d$ onto V is

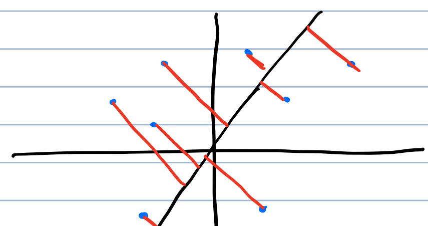
$$\text{proj}_V(X) = \sum_{j=1}^{k} \langle X, v_j \rangle \cdot v_j,$$

$$\|\text{proj}_V(X)\|_2^2 = \langle \text{proj}_V(X), \text{proj}_V(X) \rangle$$

$$= \sum_{j,l} \langle X, v_j \rangle \langle X, v_l \rangle \langle v_j, v_l \rangle = \sum_{j=1}^{k} \langle X, v_j \rangle^2$$

PCA: What is the k-dimensional subspace that explains the most variance in the dataset?

e.g. $k = 1$.



Given principal components $v_1, \cdots, v_k$, we can approximate data points with their projection:

$$X_i \approx \text{proj}_V(X_i) = \sum c_{ij} v_j$$

Since v_j are orthonormal basis, we can rewrite the projection in this basis as a k-dimensional vector

$$\text{proj}_V(X_i) = \begin{pmatrix} c_{i1} \\ c_{i2} \\ \vdots \\ c_{ik} \end{pmatrix}$$

component of X_i along the 2nd PC.

Some structural facts:

The PCs are not always unique!

However, if they are unique, then the solutions are nested!

$k = 1 \to \text{span}(\{v_1\})$

$k = 2 \to \text{span}(\{v_1, v_2\})$

$k = 3 \to \text{span}(\{v_1, v_2, v_3\})$

we'll see why next lecture!

Next lecture: there are efficient algorithms for PCA using connections to singular value decomposition.

Relationship to Johnson Lindenstrauss

JL also gives a low-dimensional representation of data.

PCA	JL
- doesn't preserve distances	- preserves distances
- data dependent	- data oblivious
- PCs are meaningful	- JL directions are random → not meaningful.

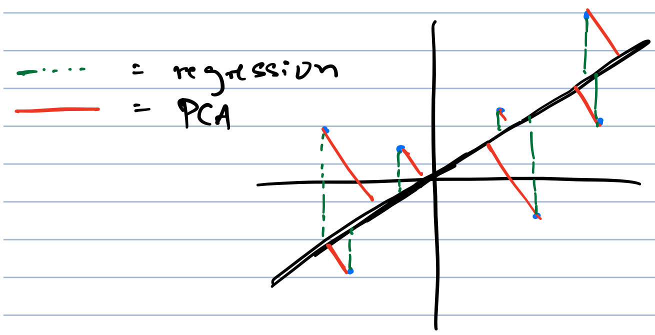
Relationship with linear regression

Regression is a way to explain one dependent variable using data.

$$\underbrace{(\overset{\in \mathbb{R}^d}{x_1}, \overset{\in \mathbb{R}}{y_1})}_{\in \mathbb{R}^{d+1}}, \ldots, (x_n, y_n)$$

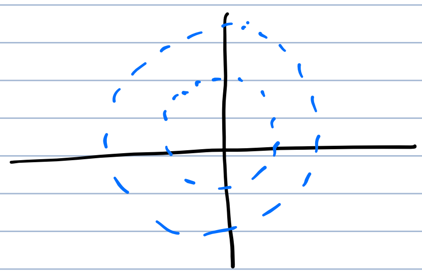
$$y \approx \langle \theta, x \rangle$$

even in 2D is a bit different:



Failure modes of PCA

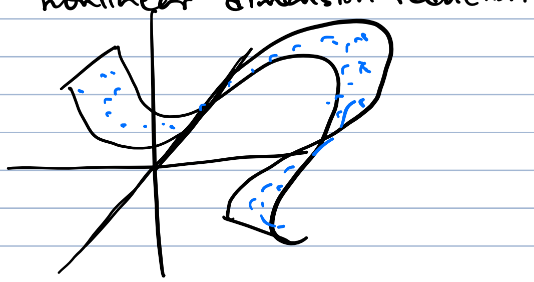
Main issue: can only discover linear structure.



A natural idea: kernelize data!

Related concept: "manifold learning"

"nonlinear dimension reduction"



Another visualization tool

t-SNE

not linear but tries to find low-d representation that "looks" like original data.