Removing Redundancies From Data: Principle Component Analysis

COMS21202, Part III

- Song Liu (song.liu@bristol.ac.uk), Lecturer in Data Science and A.I.

Objectives

- Understand potential harm of high dimensionality of dataset
- Use Principle Component Analysis (PCA) to remove “redundant” dimensions from data.

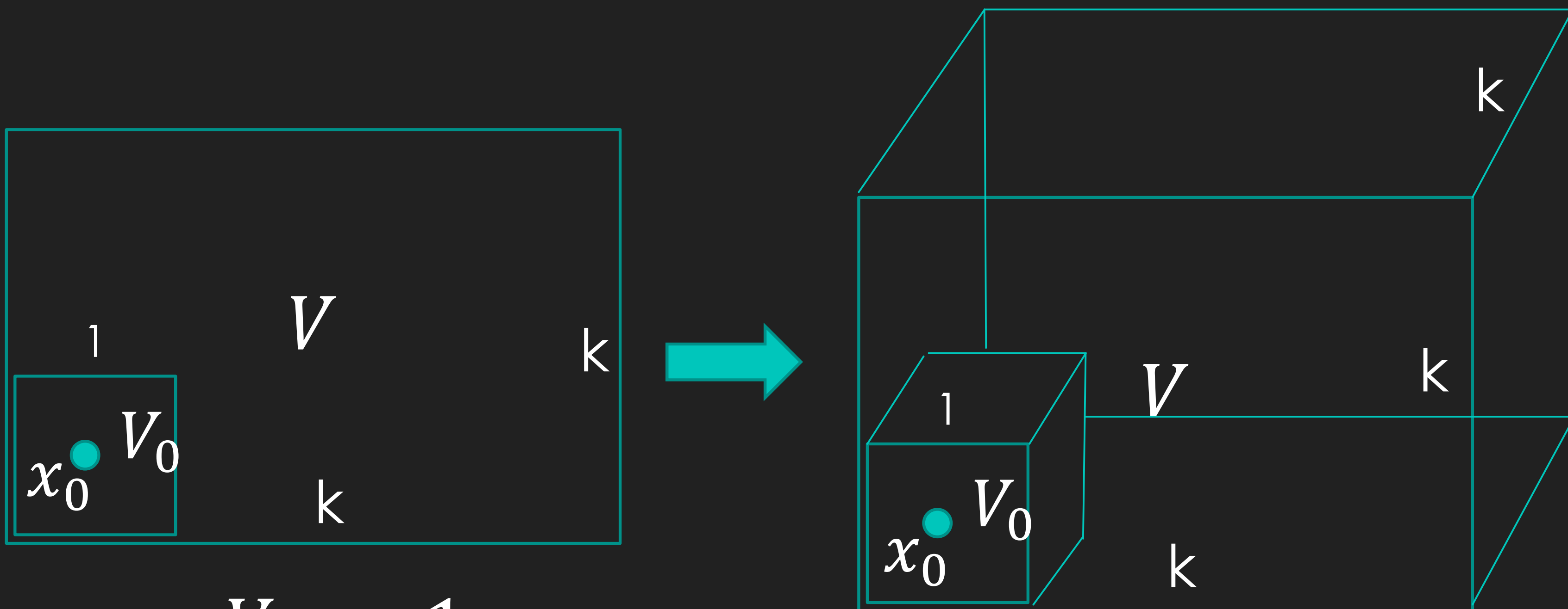
High Dimensionality, Good? Bad?

- $X = \{x_i\}_{i=1}^{n}, x \in R^d$.
- Is a large d always a good thing?
  - ☺ We have more info as d grows!
  - ☹ LS does not work when $d > n$
  - ☹ Large d causes overfitting
  - More ☹ ?

Curse of Dimensionality (CoD)

- CoD is a generic term referring to the fact that many machine learning algorithms scale very poorly with d, in terms of performance.
  - **Reason**: Many geometry concepts can have very different meanings in lower and higher dimensional space.
  - One of those is the concept of "locality".

The Vanishing Neighborhood



$$\frac{V_0}{V} = \frac{1}{k^2}$$

$$\frac{V_0}{V} = \frac{1}{k^3}$$

V_0: Neighborhood volume of x_0

$$\lim_{d \to \infty} \frac{V_0}{V} = \frac{1}{k^d} = 0$$

- Song Liu (song.liu@bristol.ac.uk), Lecturer in Data Science and A.I.

The Vanishing Neighborhood

- The neighborhood cube quickly vanishes as d increases.
- As a result, your k-nearest neighbors are **no longer** in the neighborhood V_0.
- These neighbors are no longer good at predicting the label of x_0.

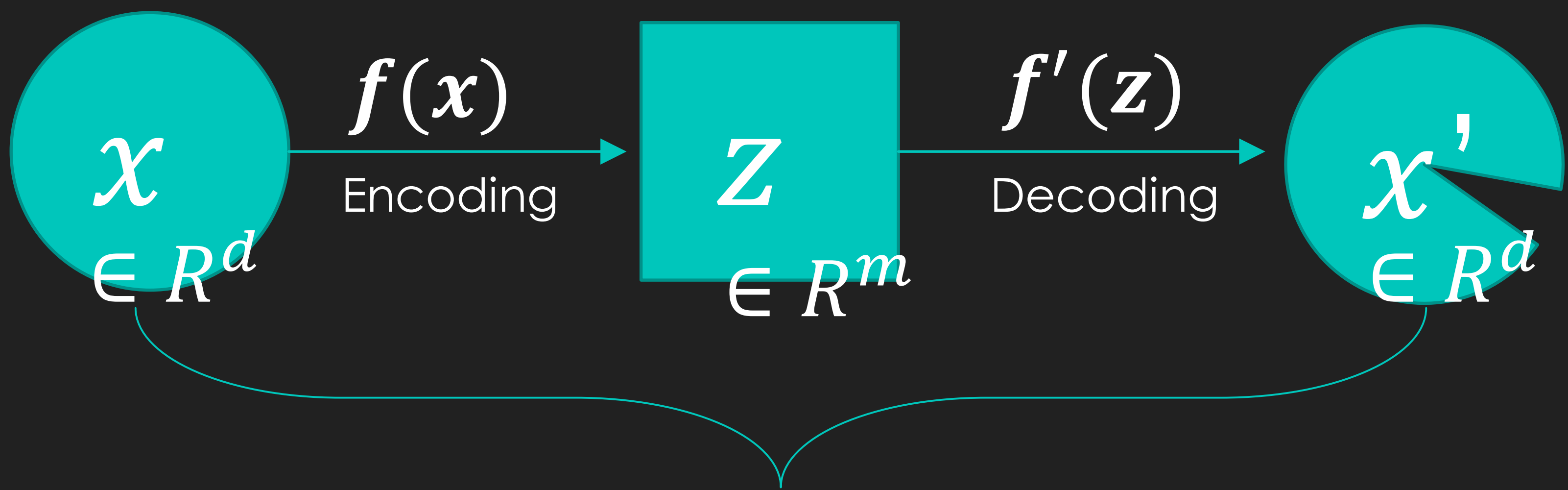
Reduce the Dimensionality using Feature Transform

- We want to find a **feature transform** $\boldsymbol{f}(\boldsymbol{x}) \in R^m$, where $m \ll d$.
  - $\boldsymbol{f}$ transforms original input $\boldsymbol{x}$ to a subspace as $R^m \subset R^d$.
  - We assume our dataset is **centered:**
    - $\frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}_i = \boldsymbol{0}$
    - **If dataset X' is not centered:**
    - **Centering:** $\forall_i\ \boldsymbol{x}_i = \boldsymbol{x}_i' - \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}'_i$

Reduce the Dimensionality using Feature Transform

- What is the optimal strategy of selecting $\boldsymbol{f(x)}$?
- Want to reduce dimension using $\boldsymbol{f}$.
  - while preserving **as much info as possible**!
- Let's look at this problem from data compression perspective!

Encoder and Decoder



Loss of Info: $||x_i - x_i'||^2$

• Song Liu (song.liu@bristol.ac.uk), Lecturer in Data Science and A.I.

Codec

Encoding

Decoding

Original recording

Music you hear

- Song Liu (song.liu@bristol.ac.uk), Lecturer in Data Science and A.I.

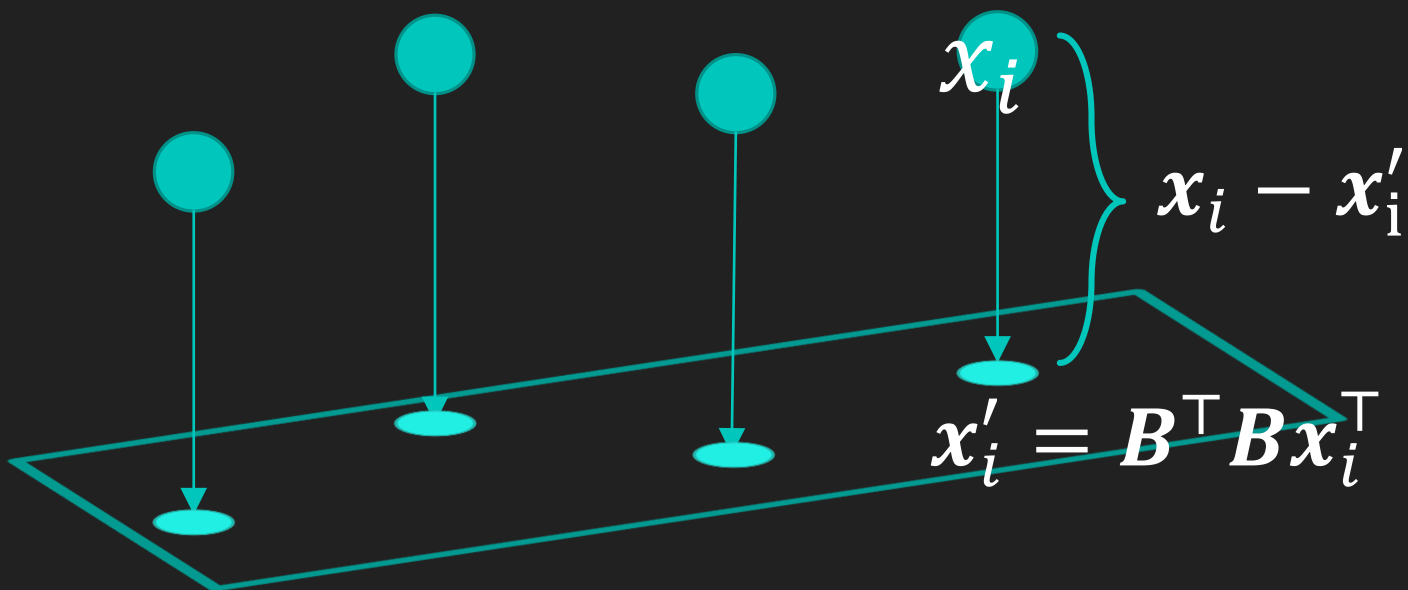
Linear Codec

- Suppose $\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{B}\boldsymbol{x}^\top$, $\boldsymbol{B} \in R^{m\times d}$.
- Suppose $\boldsymbol{f}'(\boldsymbol{z}) = \boldsymbol{B}'\boldsymbol{z}^\top$, $\boldsymbol{B}' \in R^{d\times m}$.
- We can learn a codec by
- $\min_{\boldsymbol{B},\boldsymbol{B}'} \sum_{i=1}^{n} \left|\left|\boldsymbol{x}_i^\top - \boldsymbol{B}'\boldsymbol{B}\boldsymbol{x}_i^\top\right|\right|^2$
  - However, there are so many possible candidates $\boldsymbol{B}$ and $\boldsymbol{B}'$!
  - Solving above problem is **hard**.

Linear Codec

- We need to put **constraints** on the $\boldsymbol{B}$ and $\boldsymbol{B}'$ to make our problem easier.
- One possible constraint is:
  - $\boldsymbol{B}' = \boldsymbol{B}^\top$
  - $\boldsymbol{B}\boldsymbol{B}' = \boldsymbol{B}\boldsymbol{B}^\top = \boldsymbol{I}$
- Such a codec actually defines an **orthogonal projection of $\boldsymbol{X}$.**
  - Show $\boldsymbol{B}'\boldsymbol{B}$ is an orth. projection matrix

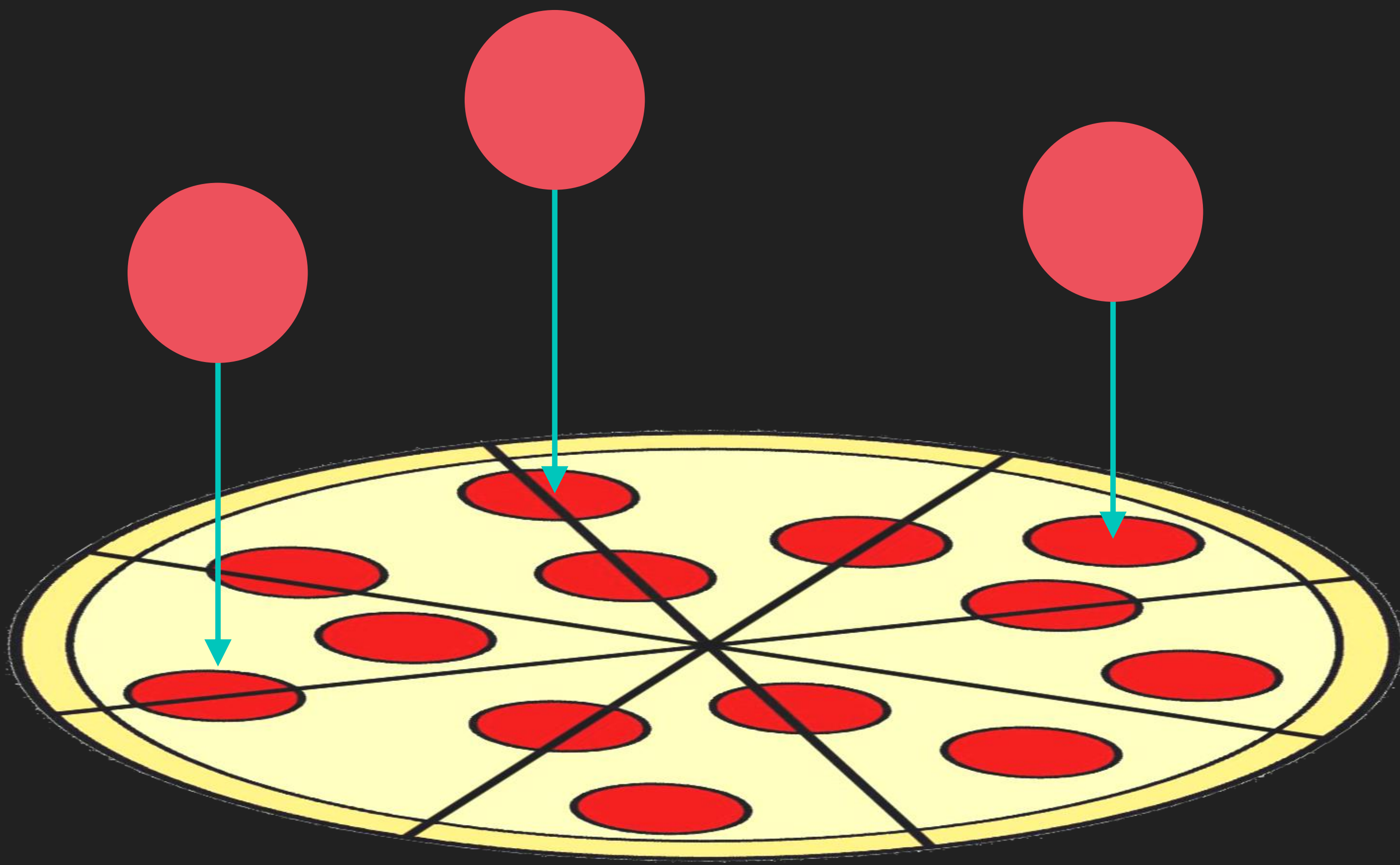
Orthogonal Projection



$z_i = f(x_i) = B x_i^\top$ is called an **embedding** of x_i, B is called embedding matrix.

- Song Liu (song.liu@bristol.ac.uk), Lecturer in Data Science and A.I.

A Pizza Topping Analogy of Embedding



- Song Liu (song.liu@bristol.ac.uk), Lecturer in Data Science and A.I.

Minimizing Projection Error

- $\min_{\boldsymbol{B}, \boldsymbol{B}\boldsymbol{B}^\top = \boldsymbol{I}} \sum_{i=1}^{n} \left\| \boldsymbol{x}_i^\top - \boldsymbol{B}^\top \boldsymbol{B} \boldsymbol{x}_i^\top \right\|^2$
  - We minimize square error between original data points and its projection.
- The above problem is equivalent to:
  - $\max_{\boldsymbol{B}, \boldsymbol{B}\boldsymbol{B}^\top = \boldsymbol{I}} \mathrm{tr}(\boldsymbol{B}\boldsymbol{X}^\top \boldsymbol{X}\boldsymbol{B}^\top)$
  - Live demonstration

Minimizing Projection Error

- $\max_{B, BB^\top = I} \operatorname{tr}(\boldsymbol{B}\boldsymbol{X}^\top \boldsymbol{X}\boldsymbol{B}^\top)$
- Remarkably, this seemingly complex optimization has an analytical solution:
- Let $[(\lambda_1, \boldsymbol{v}_1), \dots, (\lambda_m, \boldsymbol{v}_m)]$ be **sorted** eigenvalue and eigenvec of $\boldsymbol{X}^\top \boldsymbol{X}$.
  - $\lambda_1 \geq \lambda_2 \dots \geq \lambda_m$
  - $\widehat{\boldsymbol{B}} = [\boldsymbol{v}_1, \boldsymbol{v}_2, \dots, \boldsymbol{v}_m]^\top$ is an optimal solution, suppose $\boldsymbol{v}_i$ is a **column vector**.

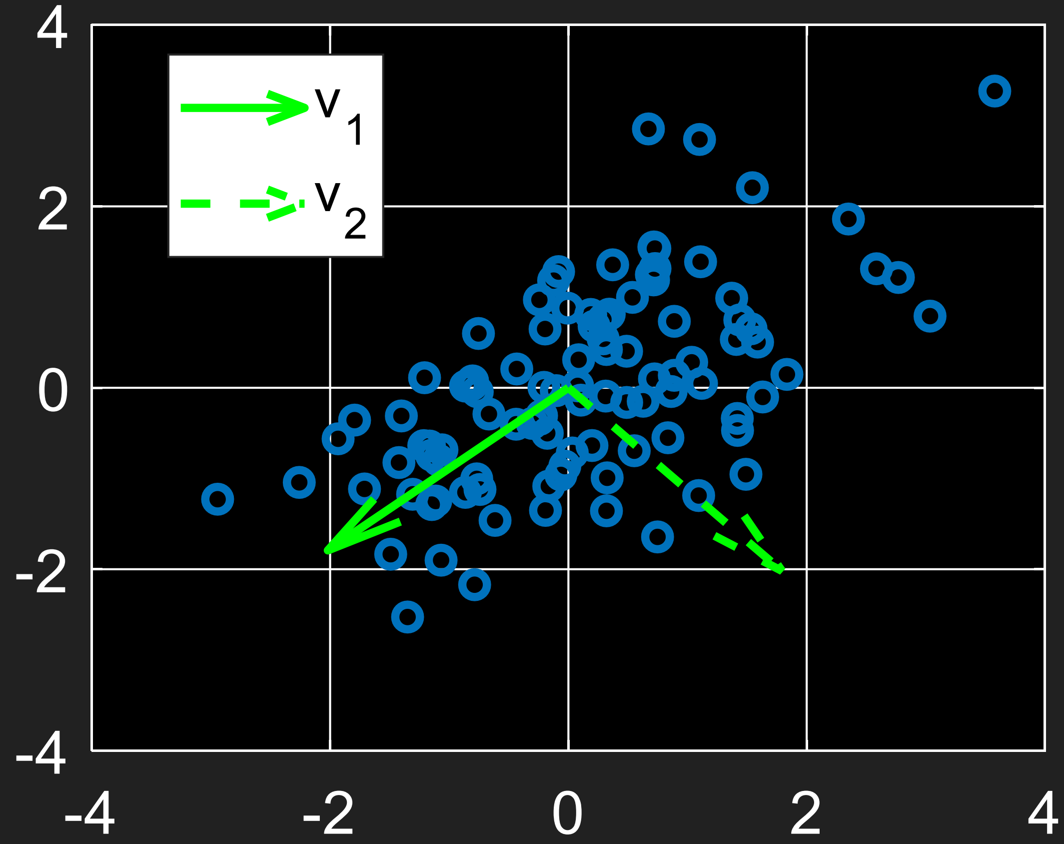
Principle Component Analysis

- As $\boldsymbol{X}$ is a centered dataset,
  - $\boldsymbol{X}^\top \boldsymbol{X} = n \cdot \mathrm{cov}[\boldsymbol{x}]$ (PC: show it!)
- Computing $\widehat{\boldsymbol{B}}$ via computing sorted eigenvectors of $\mathrm{cov}[\boldsymbol{x}]$ is called Principle Component Analysis (PCA).
- Finally, embedding $\hat{f}(\boldsymbol{x}_i) = \widehat{\boldsymbol{B}}\boldsymbol{x}_{\boldsymbol{i}}^\top \in R^m$ is called **PCA embedding** of $\boldsymbol{x}_i$.
  - m dimensional "compression" we want!

Refresh: Eigenvectors and Eigenvalues

- Given a square $n \times n$ matrix $\boldsymbol{A}$, If there exists non-zero vector $\boldsymbol{v}$ such that
- $\boldsymbol{Av} = \lambda\boldsymbol{v}, \boldsymbol{v} \in \boldsymbol{R}^n$
- Then λ is an eigenvalue and $\boldsymbol{v}$ is an eigenvector of $\boldsymbol{A}$.

Example, One Cluster

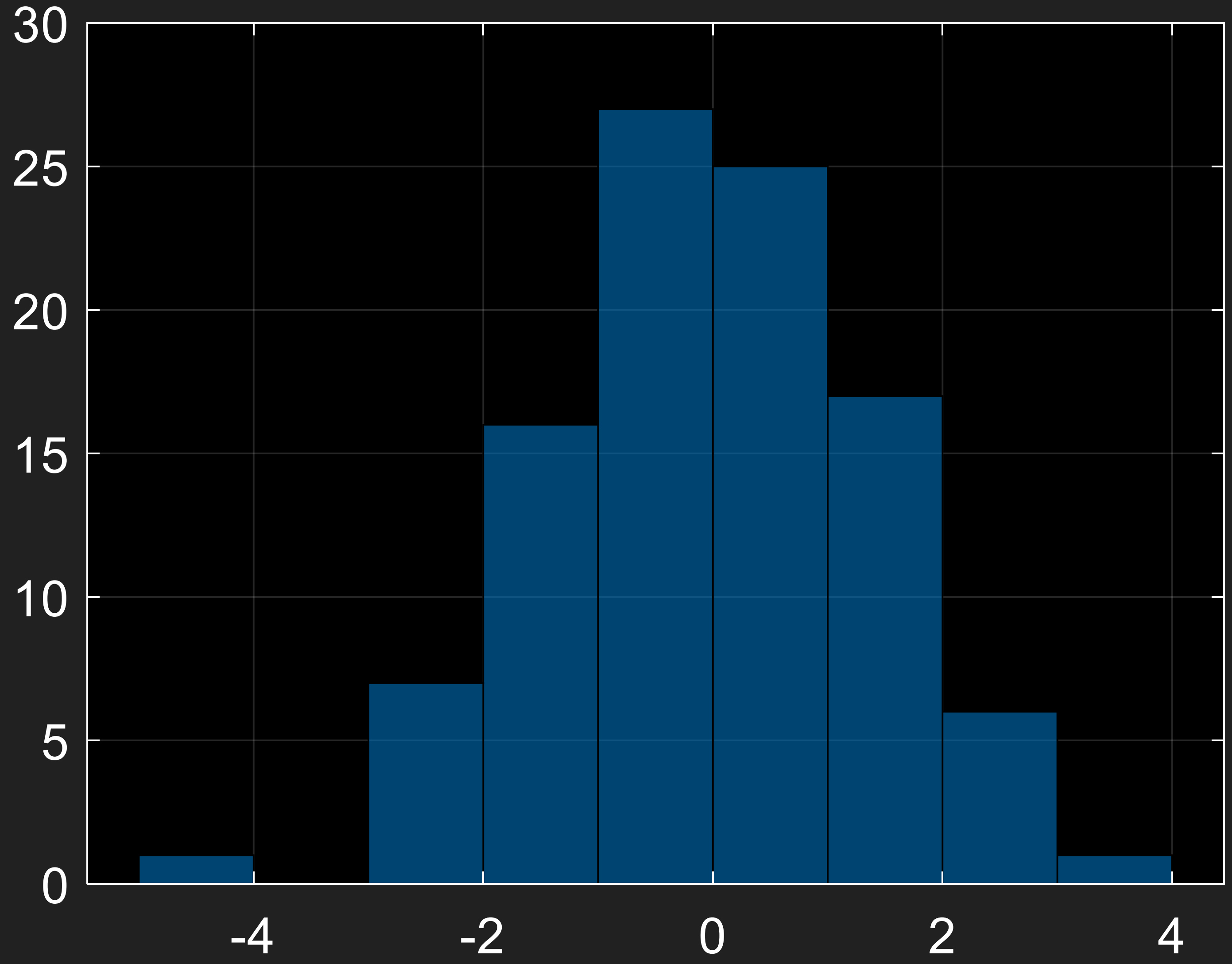


v_1 always points at the direction where your dataset has the largest variance!

PC: Intuitively explain why.

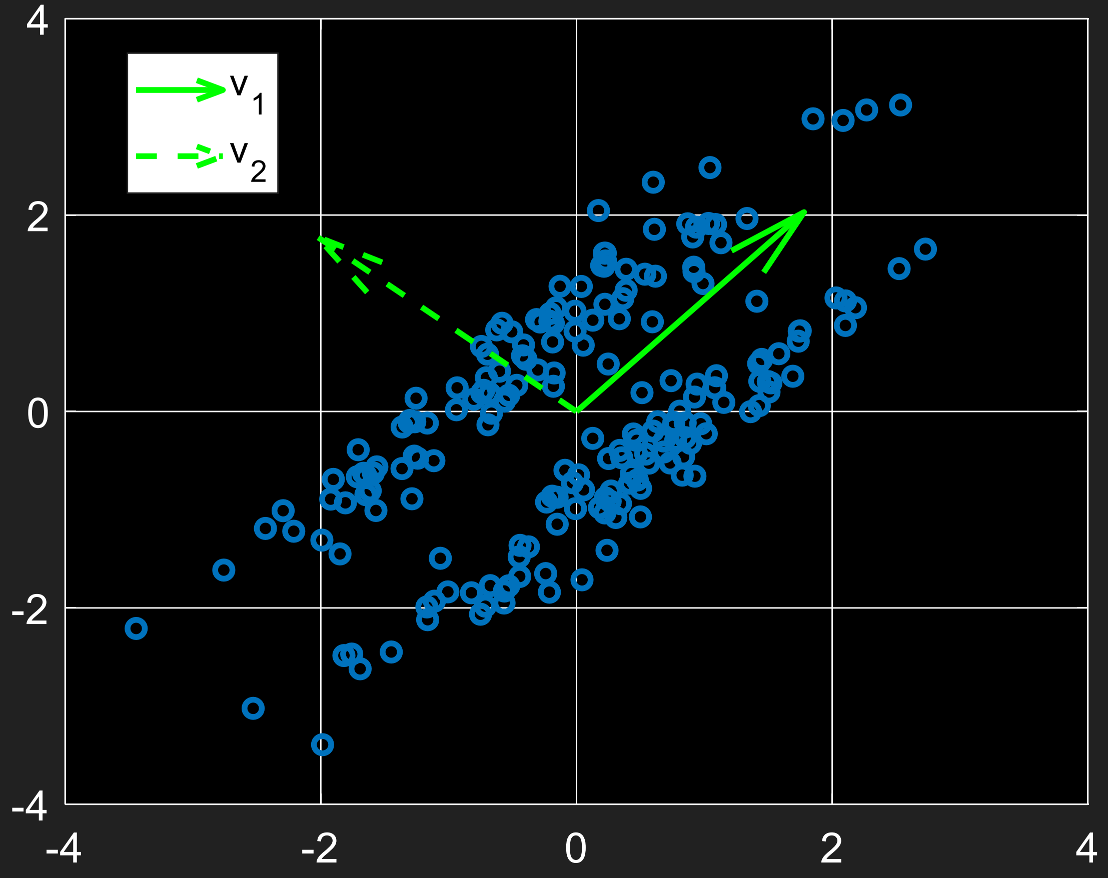
- Song Liu (song.liu@bristol.ac.uk), Lecturer in Data Science and A.I.

Example, Embedding $v_1^\top x^\top$



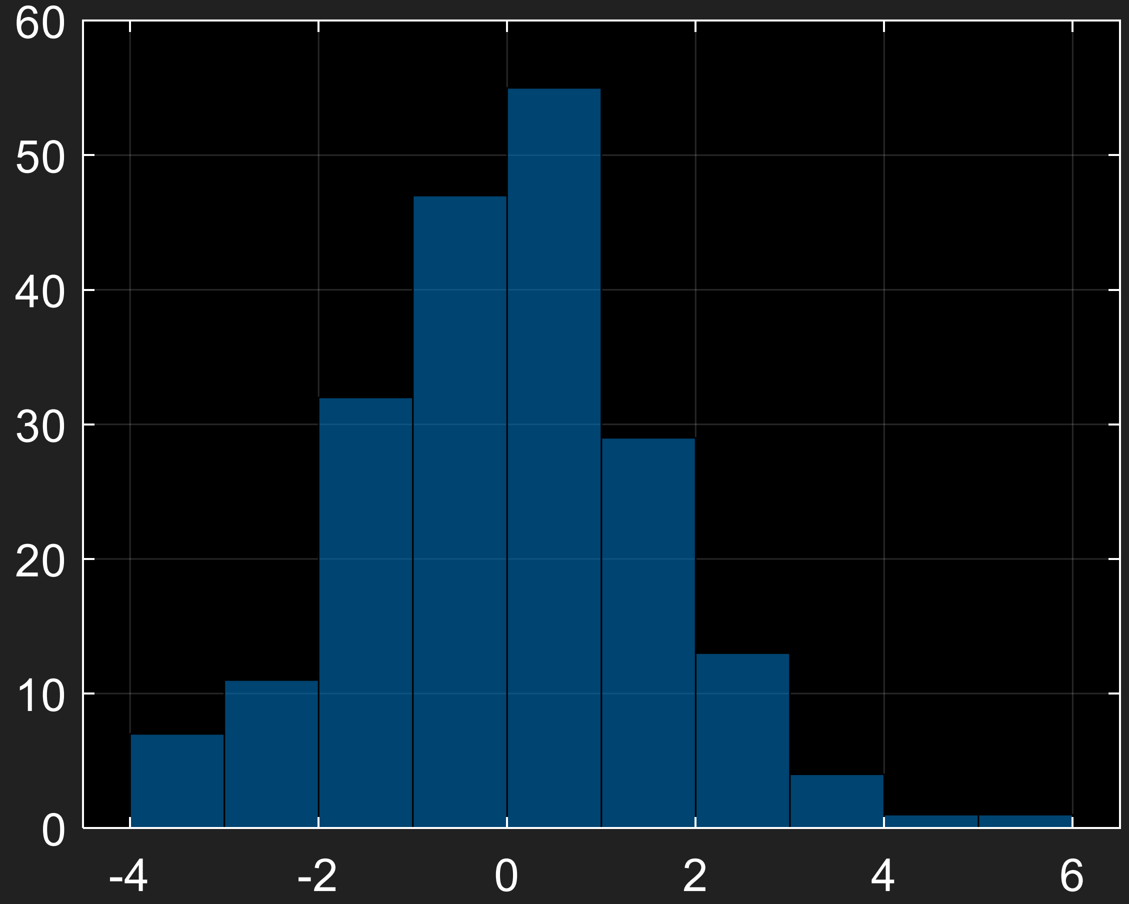
- Song Liu (song.liu@bristol.ac.uk), Lecturer in Data Science and A.I.

Example, Two Clusters



However, PCA does **not necessarily** preserve clustering information.

Example, Embedding $v_1^\top x^\top$



Cluster information **lost** after embedding!
Will address this issue in the next lecture.

Conclusion

- Curse of Dimensionality
  - d increases, performance may decrease.
- Principle Component Analysis
  - Finding an embedding matrix $\widehat{\boldsymbol{B}}$ by computing sorted eigenvalue/vectors of $\mathrm{cov}[x]$.
  - PCA Embedding: $\widehat{\boldsymbol{f}}(\boldsymbol{x}_i) = \widehat{\boldsymbol{B}}\boldsymbol{x}^\top$.

- Song Liu (song.liu@bristol.ac.uk), Lecturer in Data Science and A.I.