NLP with Classification and Vector Spaces

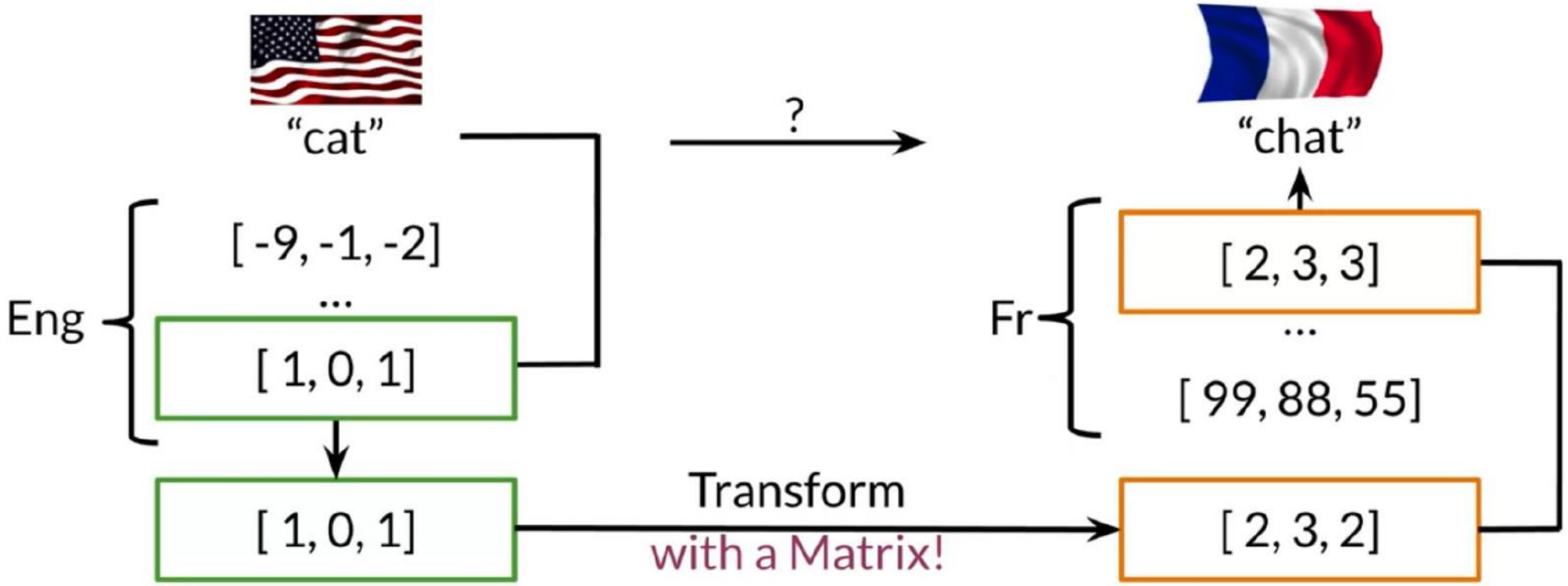
Week 4.
Machine Translation and Document Search

From *Natural Language Processing Specialization* lectures on coursera

SoYeop Yoo

01 Transforming Word Vectors

Overview of Translation



01 Transforming Word Vectors

Transforming Vectors

```
R = np.array([[2,0],
              [0,-2]])

x = np.array([[1,1]])

np.dot(x,R)
```

```
array([[2,-2]])
```

01 Transforming Word Vectors

Align Word Vectors

- $XR \approx Y$
  - X, Y
    - Subsets of the full vocabulary

$$\begin{pmatrix} \text{["cat" } vector\text{]} \\ \text{[...} vector\text{]} \\ \text{["zebra" } vector\text{]} \end{pmatrix} \qquad \begin{pmatrix} \text{["chat" } vecteur\text{]} \\ \text{[...} vecteur\text{]} \\ \text{["zebresse" } vecteur\text{]} \end{pmatrix}$$

X Y

  - R
    - Transformation matrix
    - To get R

→Optimize the distance between XR and Y by minimizing the frobenius norm

01 Transforming Word Vectors

- Solving for R

initialize R

in a loop:

$$Loss = \| \mathbf{XR} - \mathbf{Y} \|_F$$

$$g = \frac{d}{dR} Loss$$ gradient

$$R = R - \alpha g$$ update

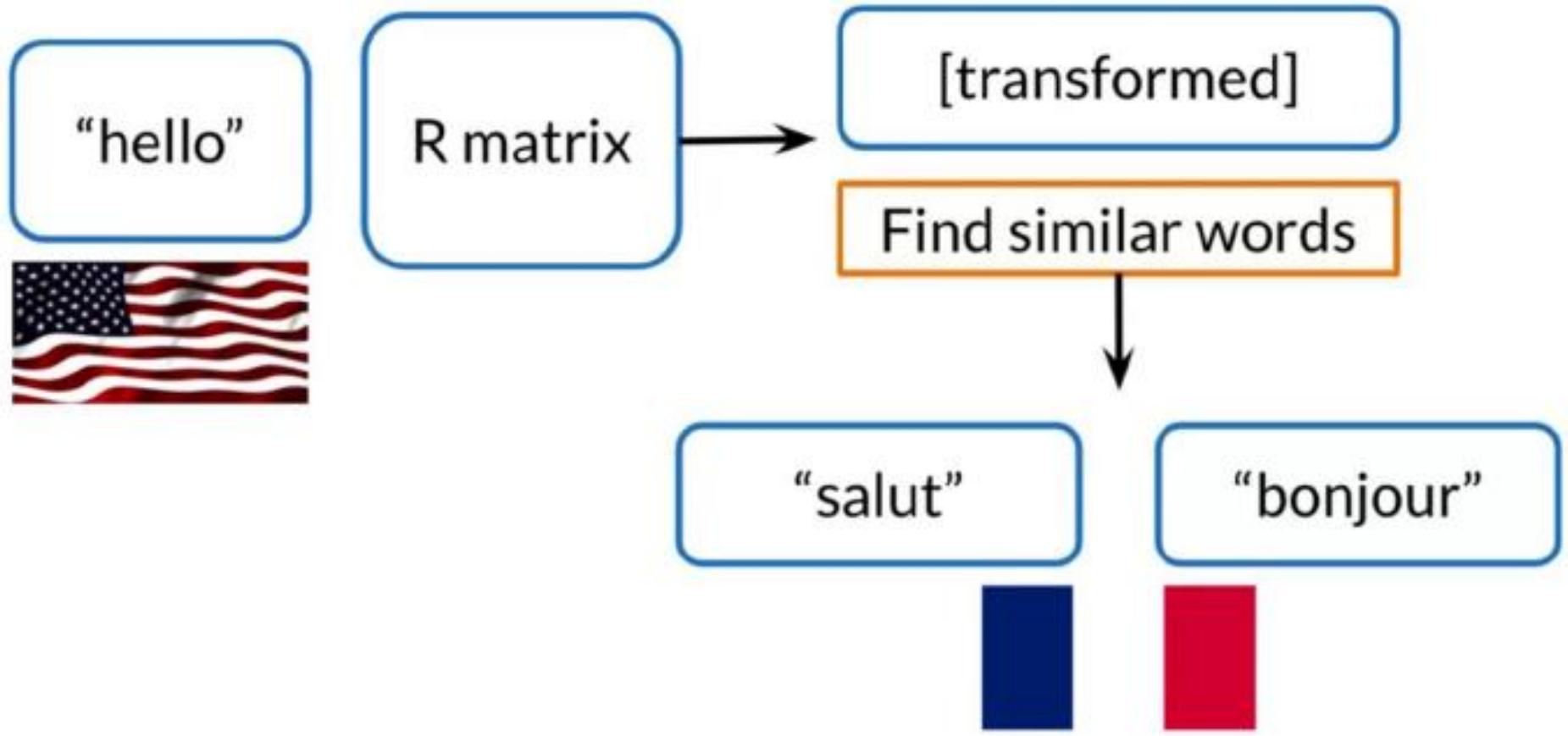
01 Transforming Word Vectors

Frobenius Norm

- Frobenius norm
  - $A = \begin{pmatrix} 2 & 2 \\ 2 & 2 \end{pmatrix}$
  - $\|A_F\| = \sqrt{2^2 + 2^2 + 2^2 + 2^2} = 4$
  - $\|A\|_F = \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} |a_{ij}|^2}$
- Frobenius norm squared
  - $A = \begin{pmatrix} 2 & 2 \\ 2 & 2 \end{pmatrix}$
  - $\|A\|_F^2 = (\sqrt{2^2 + 2^2 + 2^2 + 2^2})^2 = 16$

02 K-Nearest Neighbors

- Finding the Translation



02 K-Nearest Neighbors

Nearest Neighbors

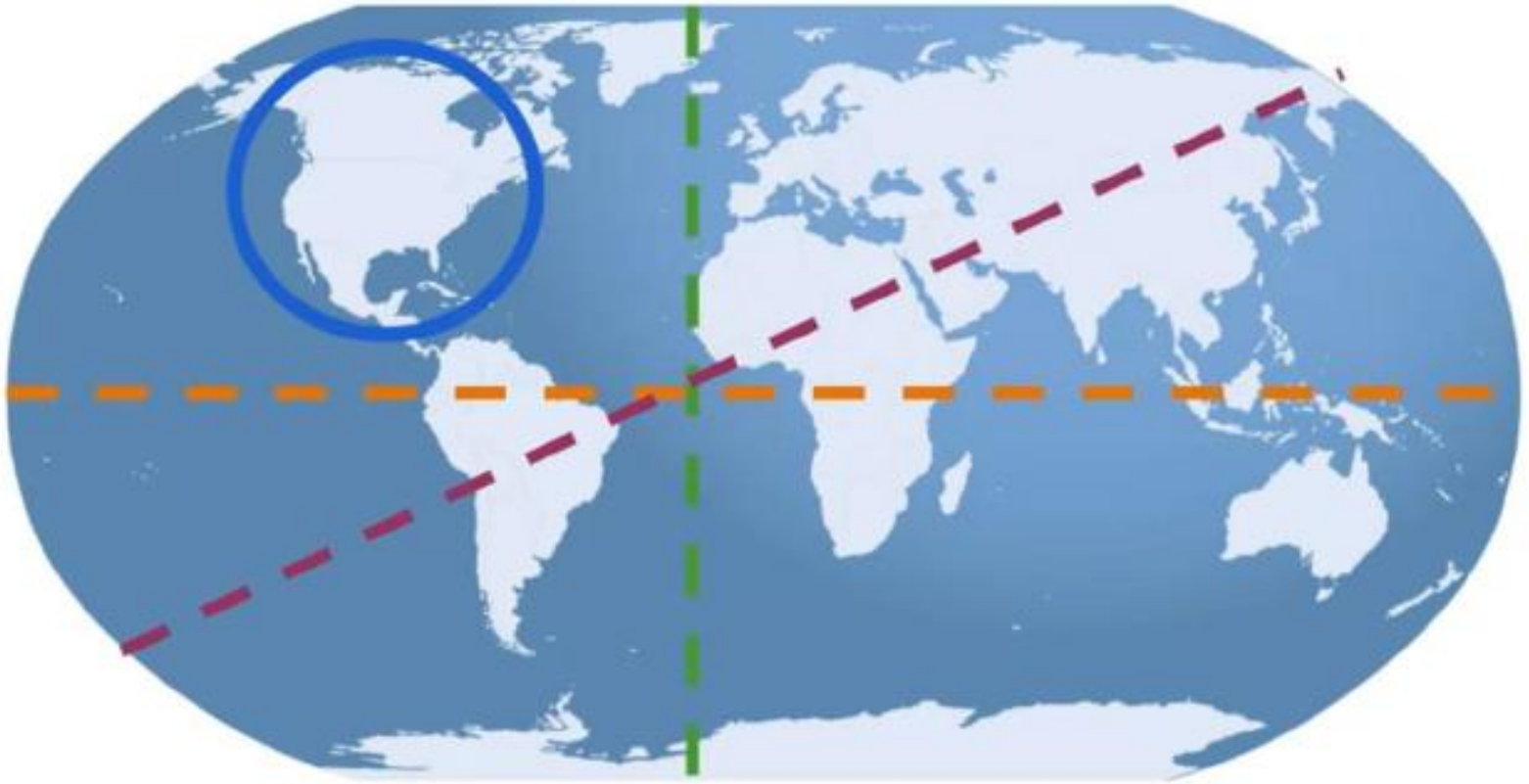
You

San Francisco

Friend	Location	Nearest
	Shanghai	2
	Bangalore	3
	Los Angeles	1

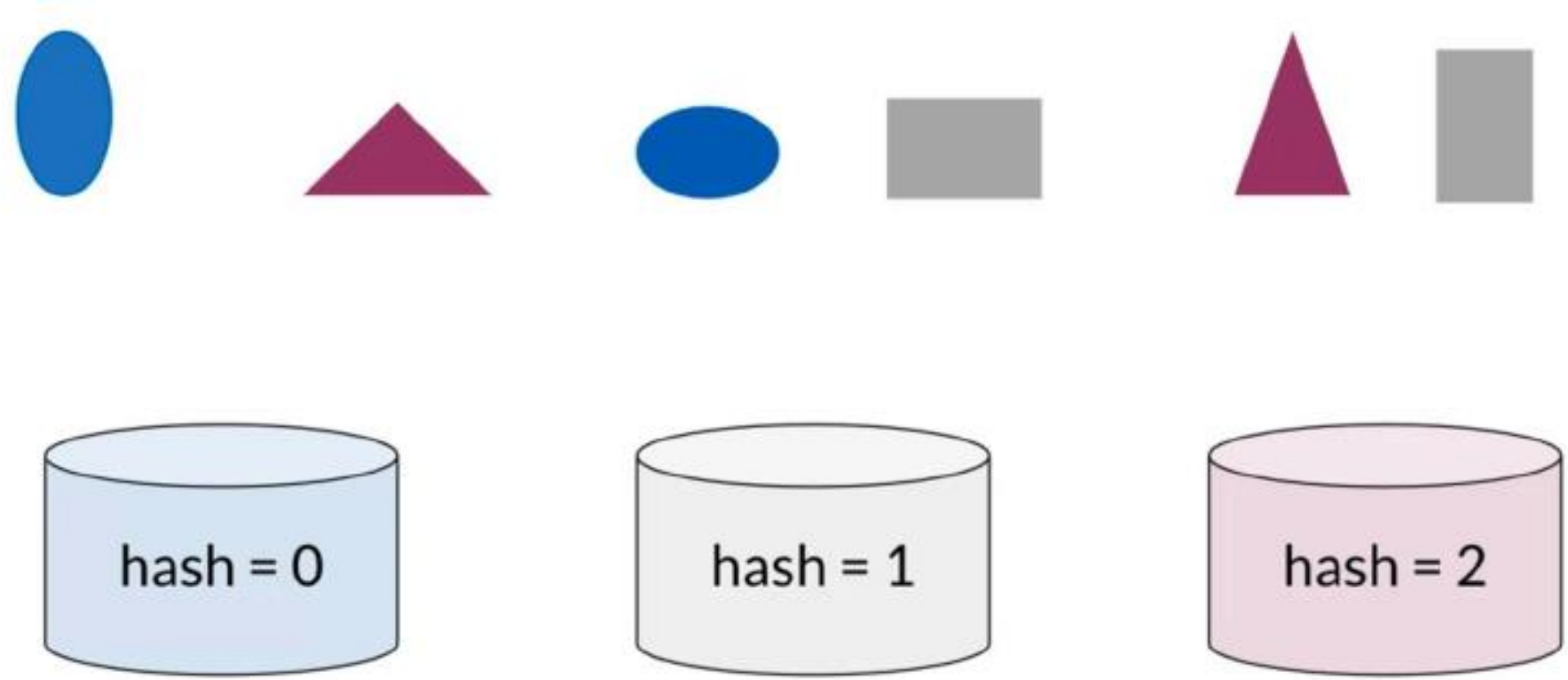
02 K-Nearest Neighbors

- Nearest Neighbors



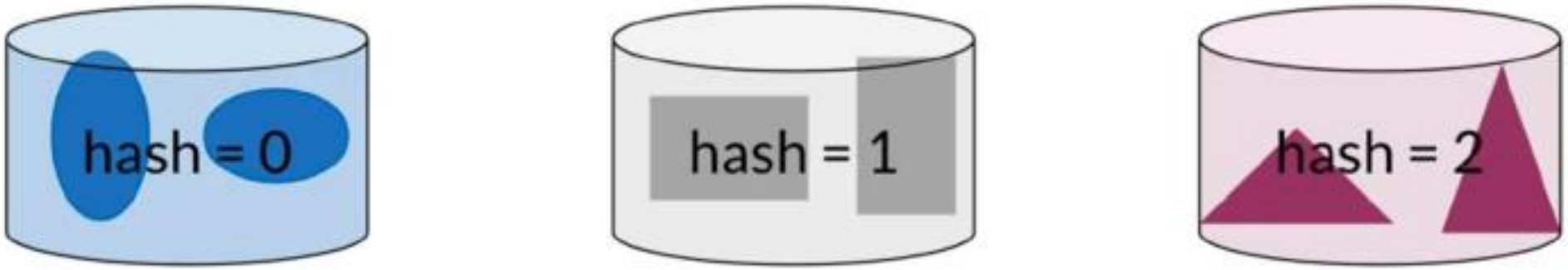
03 Hash Tables and Hash Functions

Hash Tables



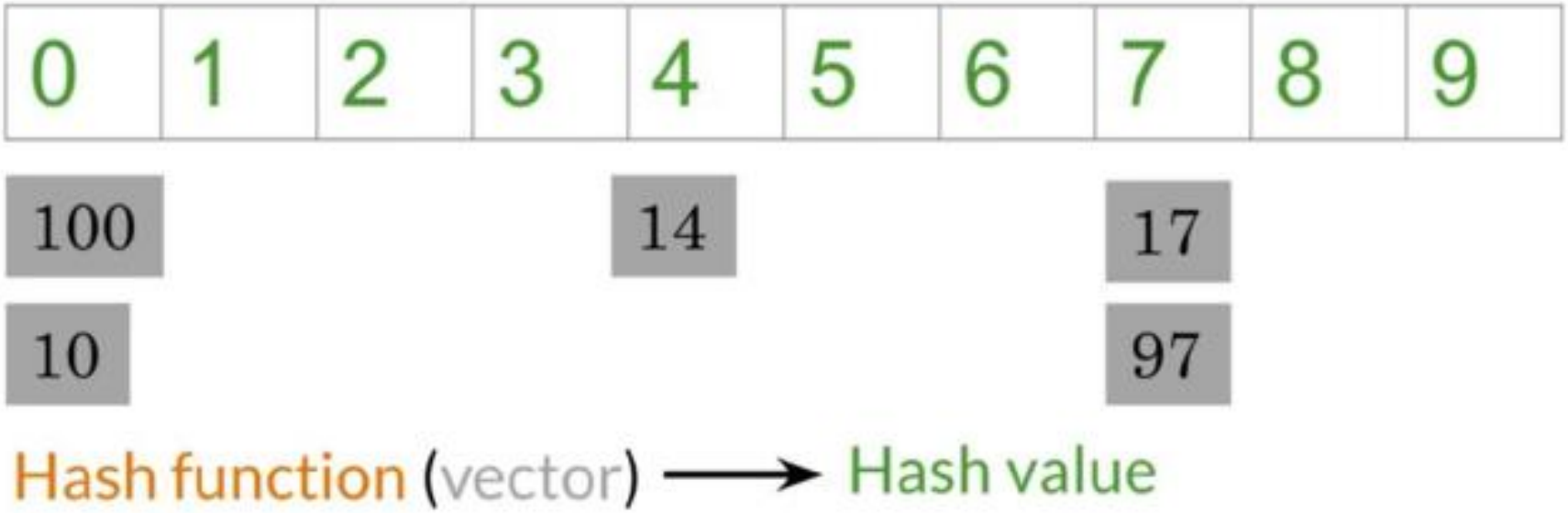
03 Hash Tables and Hash Functions

Hash Tables



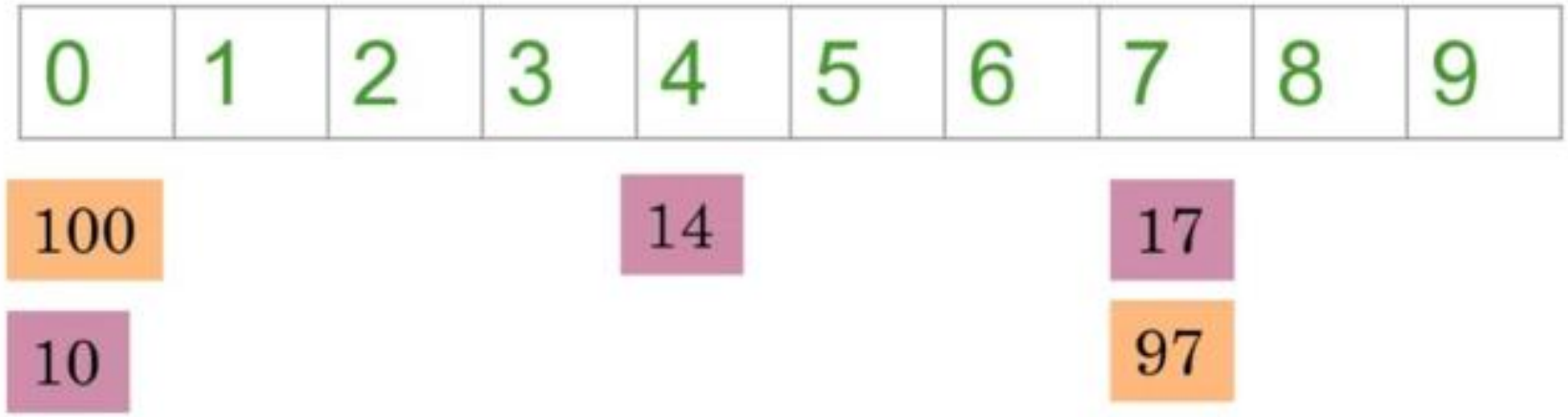
03 Hash Tables and Hash Functions

Hash Function



03 Hash Tables and Hash Functions

- Hash Function



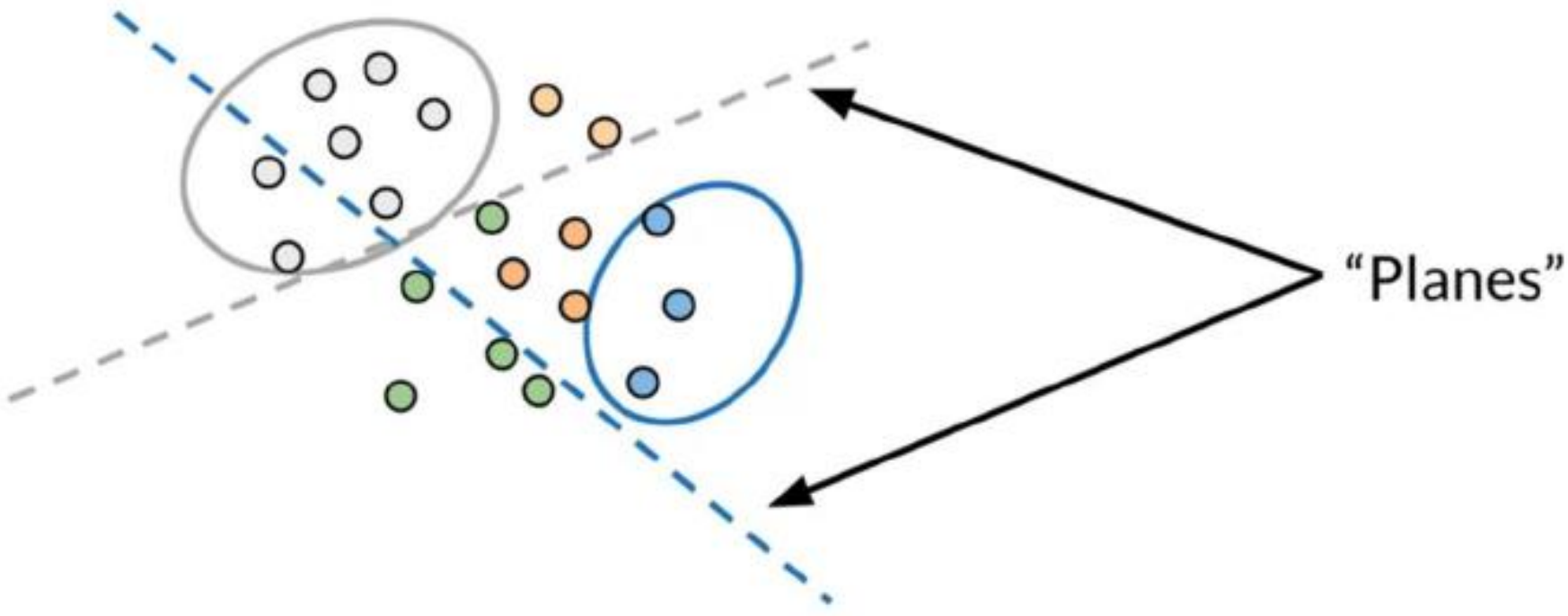
04 Locality Sensitive Hashing

Hash Function by Location

- Locality sensitive hashing
  - Hashing method that cares very deeply about assigning items based on where they're located in vector space

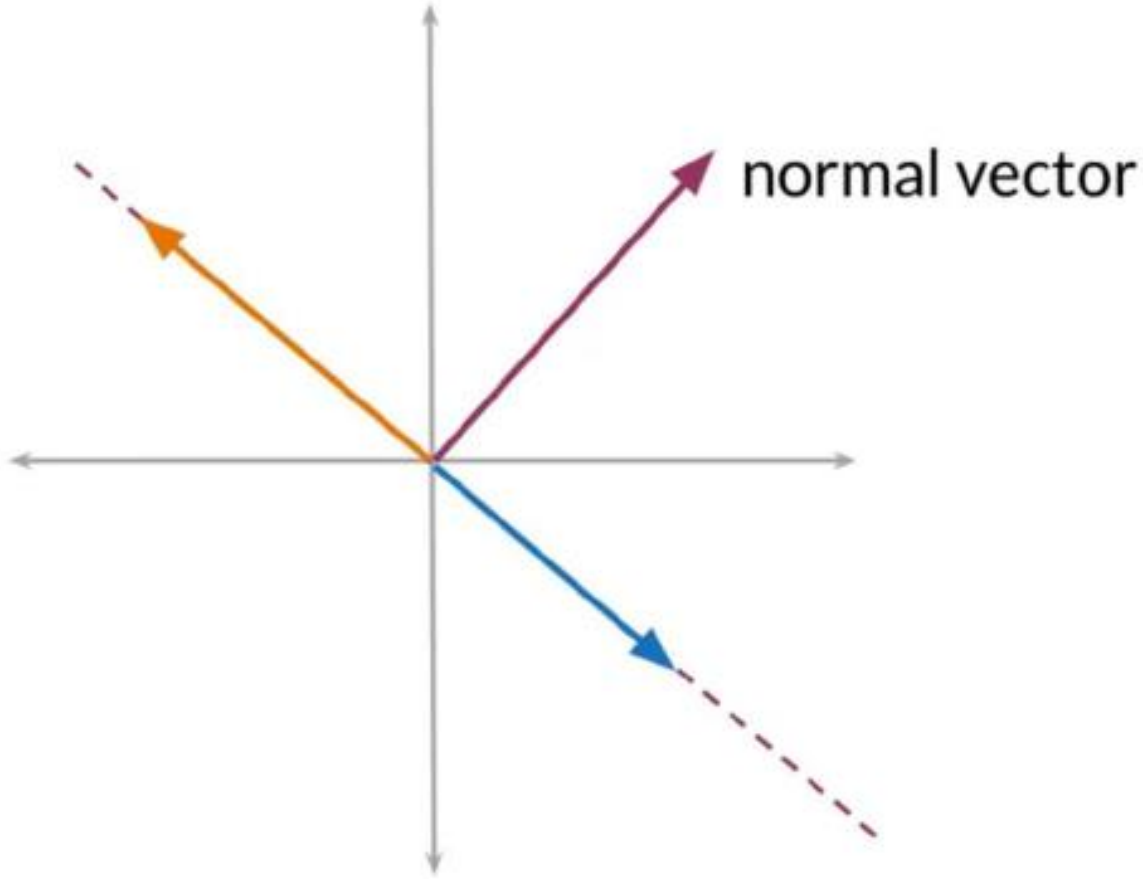
04 Locality Sensitive Hashing

Locality Sensitive Hashing



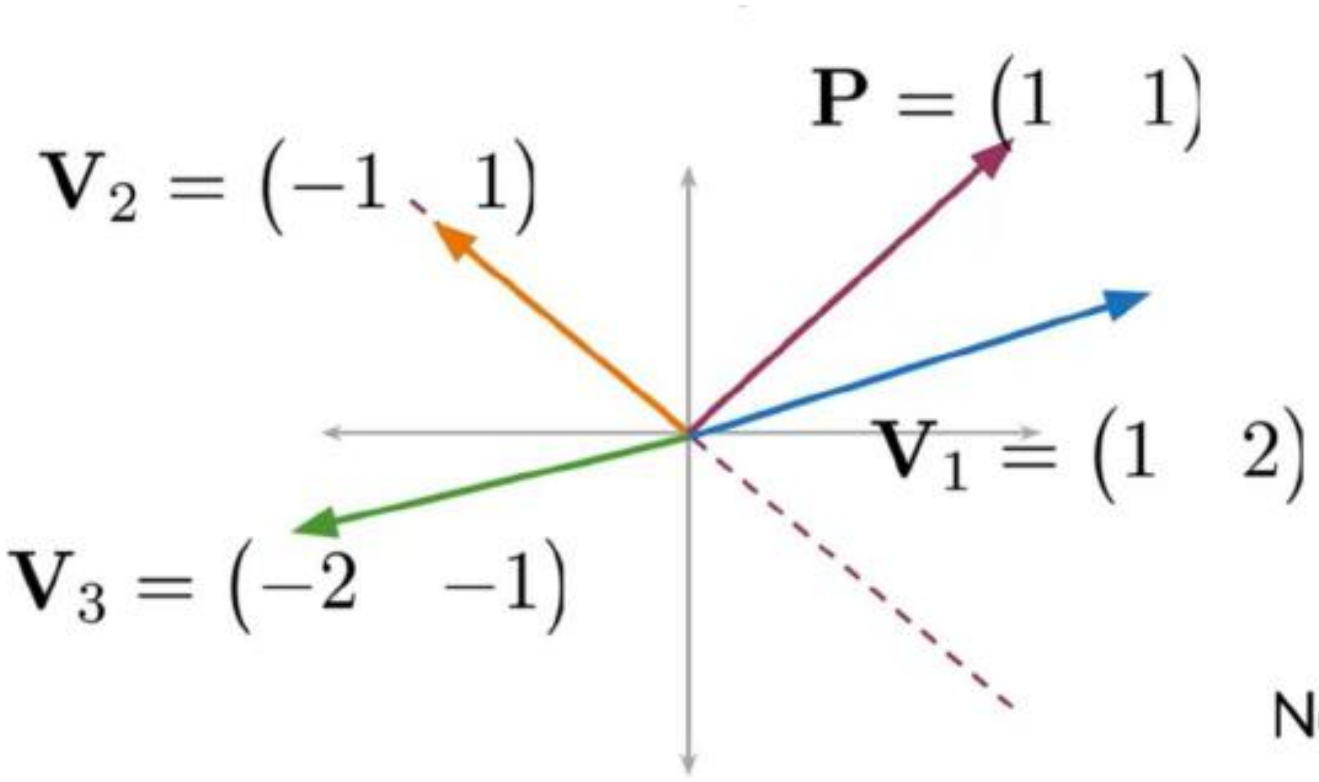
04 Locality Sensitive Hashing

- Planes



04 Locality Sensitive Hashing

- Planes

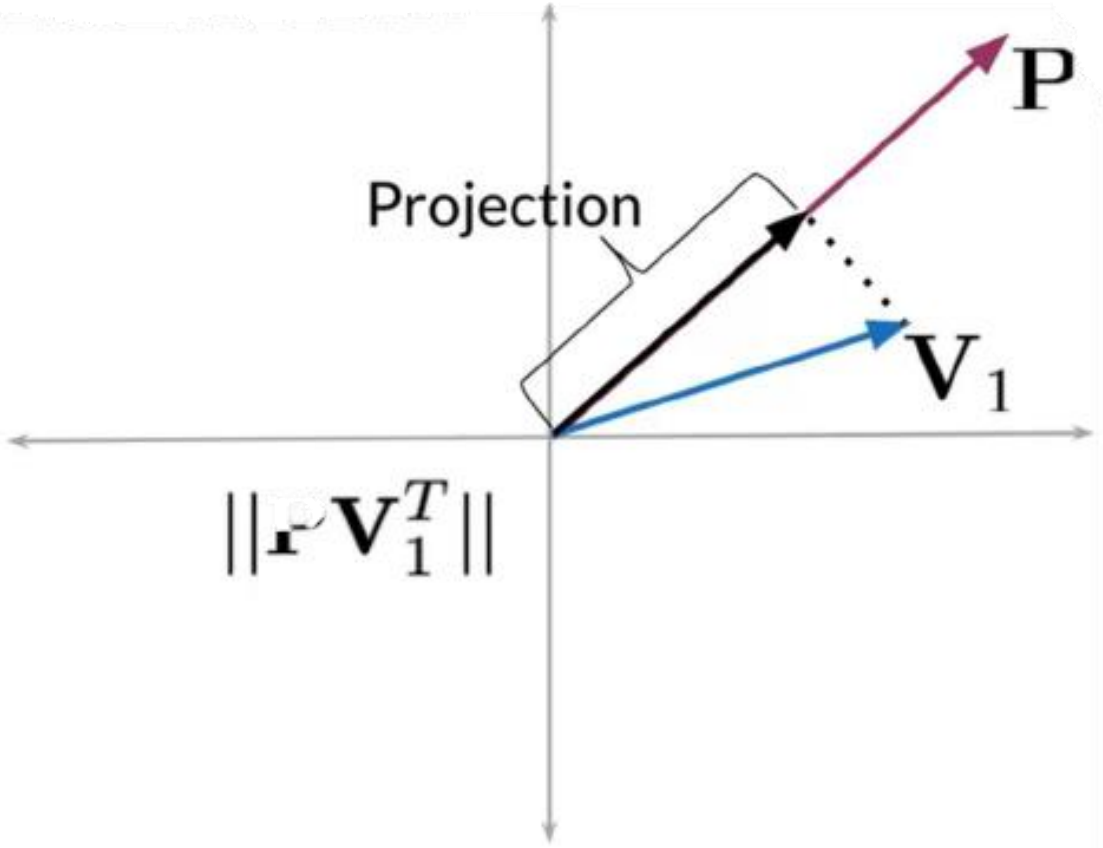


$$\mathbf{PV}_1^T = 3$$
$$\mathbf{PV}_2^T = 0$$
$$\mathbf{PV}_3^T = -3$$

Notice the signs?

04 Locality Sensitive Hashing

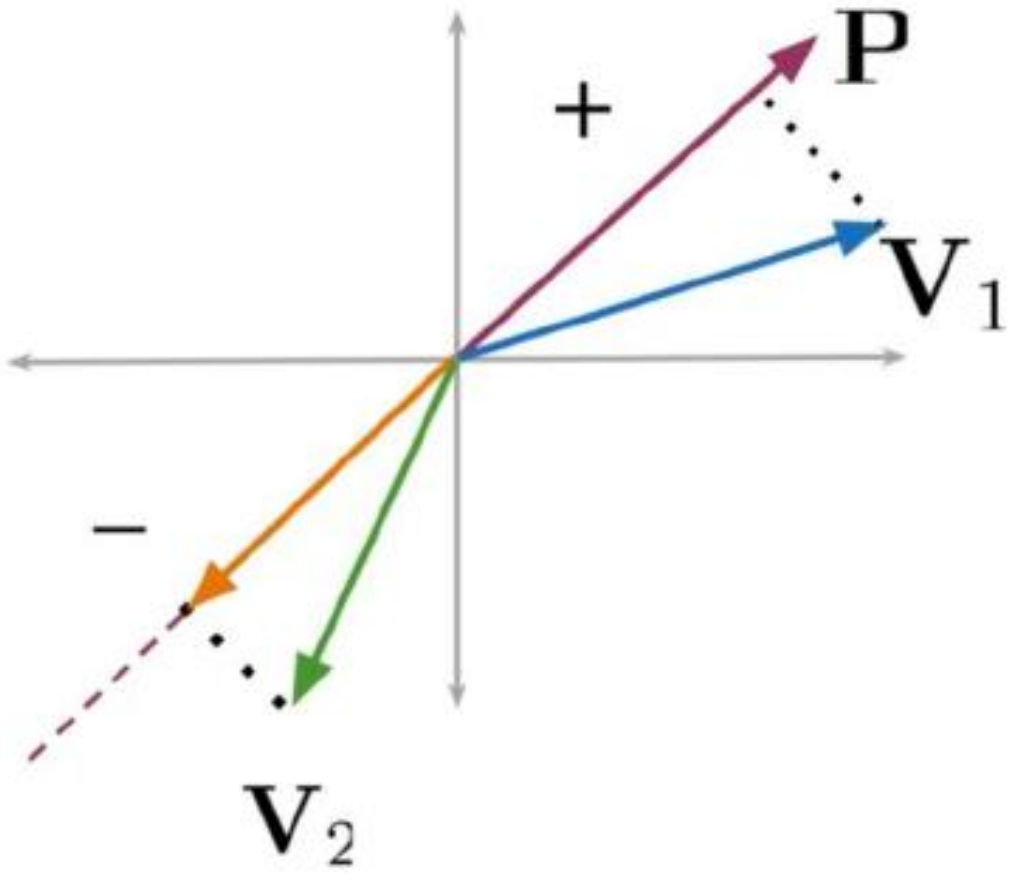
- Dop Product



04 Locality Sensitive Hashing

Dop Product

- Sign indicates direction

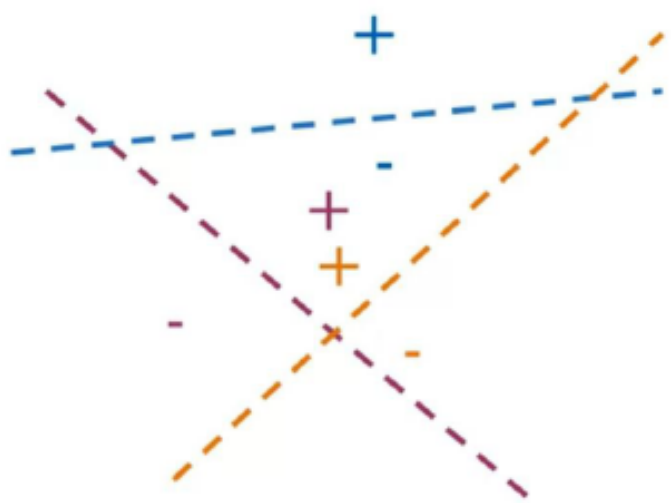


05 Multiple Planes

Multiple Planes

Single hash value

- $hash = \sum_i^H 2^i \times h_i$
  - $sign_i \geq 0 \rightarrow h_i = 1$
  - $sign_i < 1 \rightarrow h_i = 0$



$$\mathbf{P}_1\mathbf{v}^T = 3, sign_1 = +1, h_1 = 1$$

$$\mathbf{P}_2\mathbf{v}^T = 5, sign_2 = +1, h_2 = 1$$

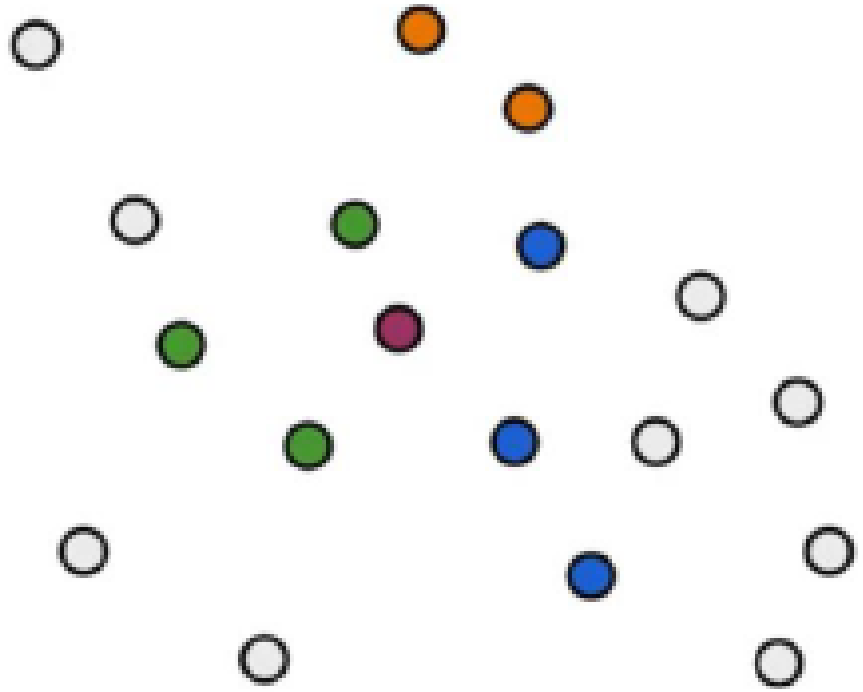
$$\mathbf{P}_3\mathbf{v}^T = -2, sign_3 = -1, h_3 = 0$$

$$hash = 2^0 \times h_1 + 2^1 \times h_2 + 2^2 \times h_3$$
$$= 1 \times 1 + 2 \times 1 + 4 \times 0$$
$$= 3$$

06 Approximated Nearest Neighbors

Multiple Sets of Random Planes

- Approximate nearest (friendly) neighbors



07 Searching Documents

Document Representation