

# California State University, Dominguez Hills

## Department of Computer Science

### CSC 595

Professor: Dr. Benyamin Ahmadnia  
[bahmadniayebosari@csudh.edu](mailto:bahmadniayebosari@csudh.edu)

Spring 2025

# Copyright Notice

These slides are distributed under the Creative Commons License.

[DeepLearning.AI](#) makes these slides available for educational purposes. You may not use or distribute these slides for commercial purposes. You may make copies of these slides and use or distribute them for educational purposes as long as you cite [DeepLearning.AI](#) as the source of the slides.

For the rest of the details of the license, see <https://creativecommons.org/licenses/by-sa/2.0/legalcode>

# Table of Contents

- Sentiment Analysis with Logistic Regression
- Sentiment Analysis with Naïve Bayes
- Vector Space Models
- Machine Translation and Document Search

# Lecture 3

## Vector Space Models



# Why Learn Vector Space?

---

Where are you heading?

Where are you from?



Different meaning

What is your age?

How old are you?



Same Meaning

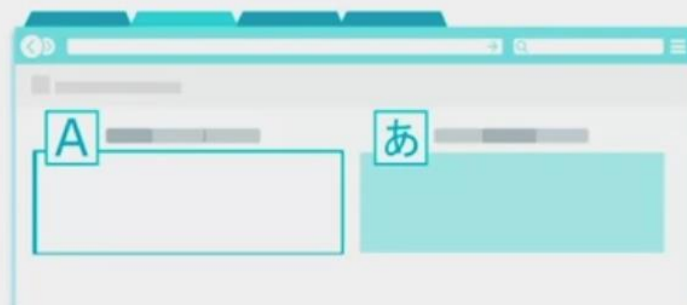
# Vector Space Models Applications

---

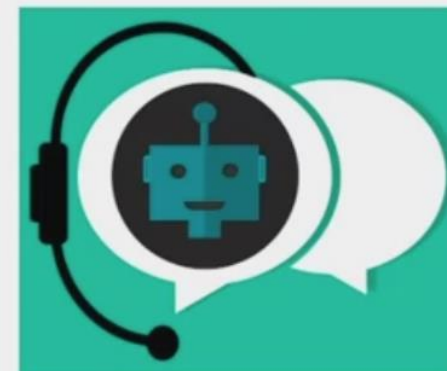
- You eat cereal from a bowl
- You buy something and someone else sells it



Information Extraction



Machine Translation



Chatbots

# Word-by-Word Design

Number of times they *occur together within a certain distance*  $k$

I like simple data

I prefer simple raw data

$k=2$

simple

data

2

# Word-by-Word Design

Number of times they *occur together within a certain distance*  $k$

I like simple data

I prefer simple raw data

$k=2$

	simple	raw	like	I
data	2	1	1	0

n

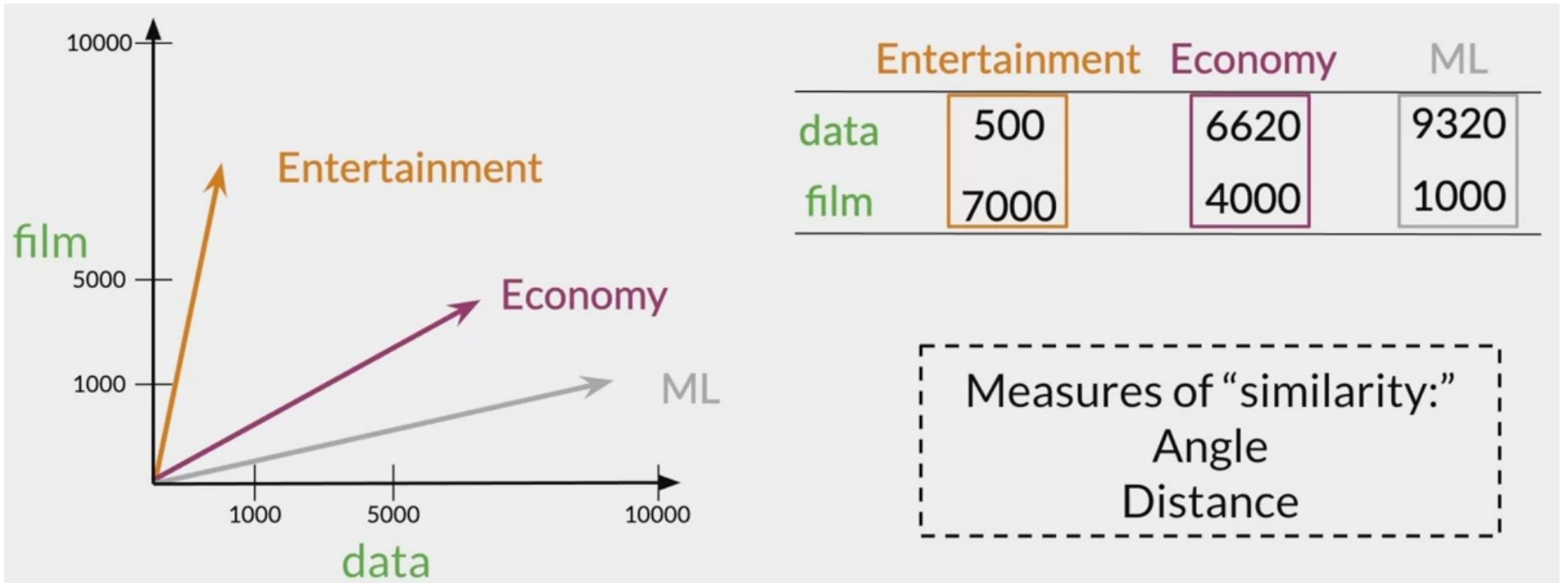


# Word-by-Document Design

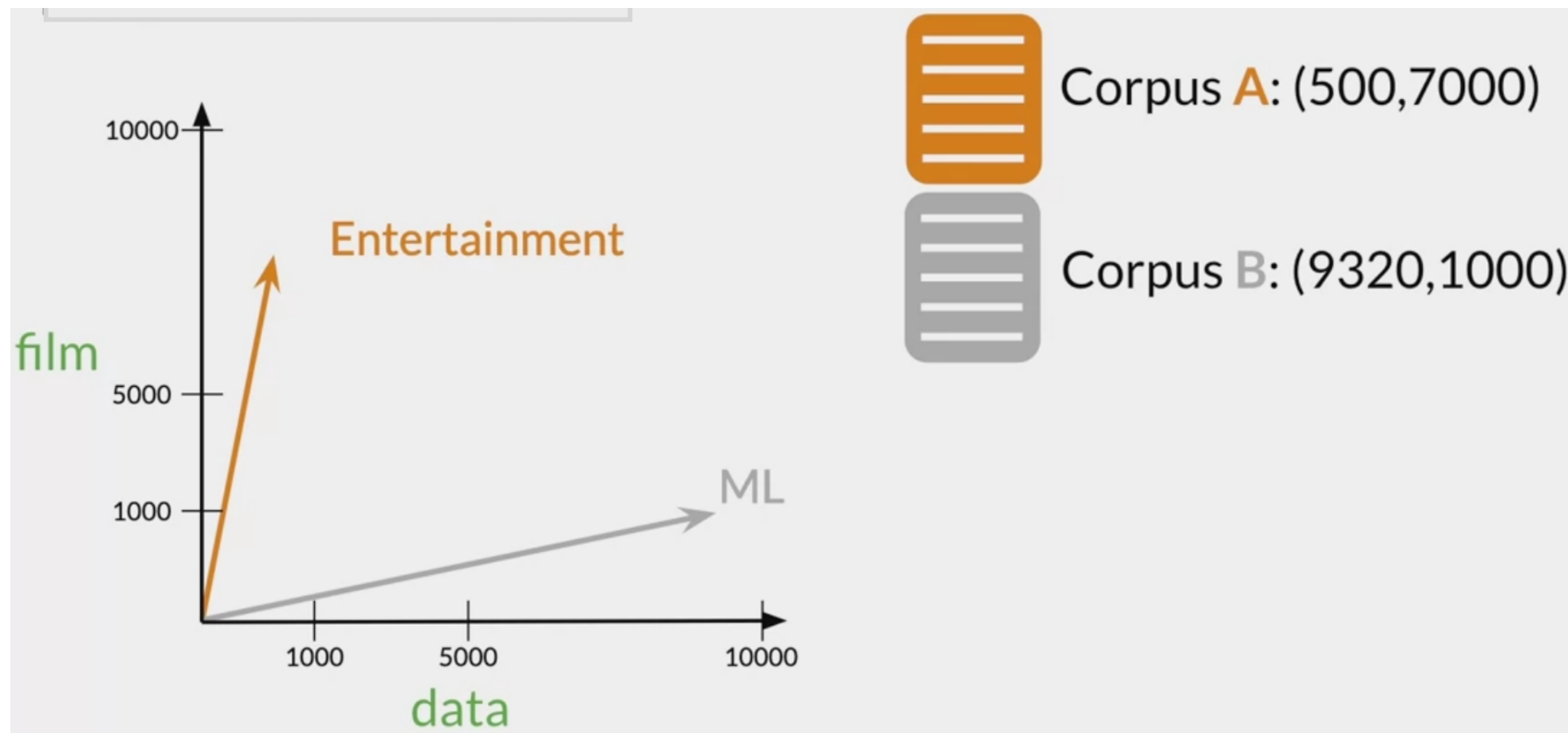
Number of times a word occurs within a certain category

	Entertainment	Economy	Machine Learning
data	500	6620	9320
film	7000	4000	1000

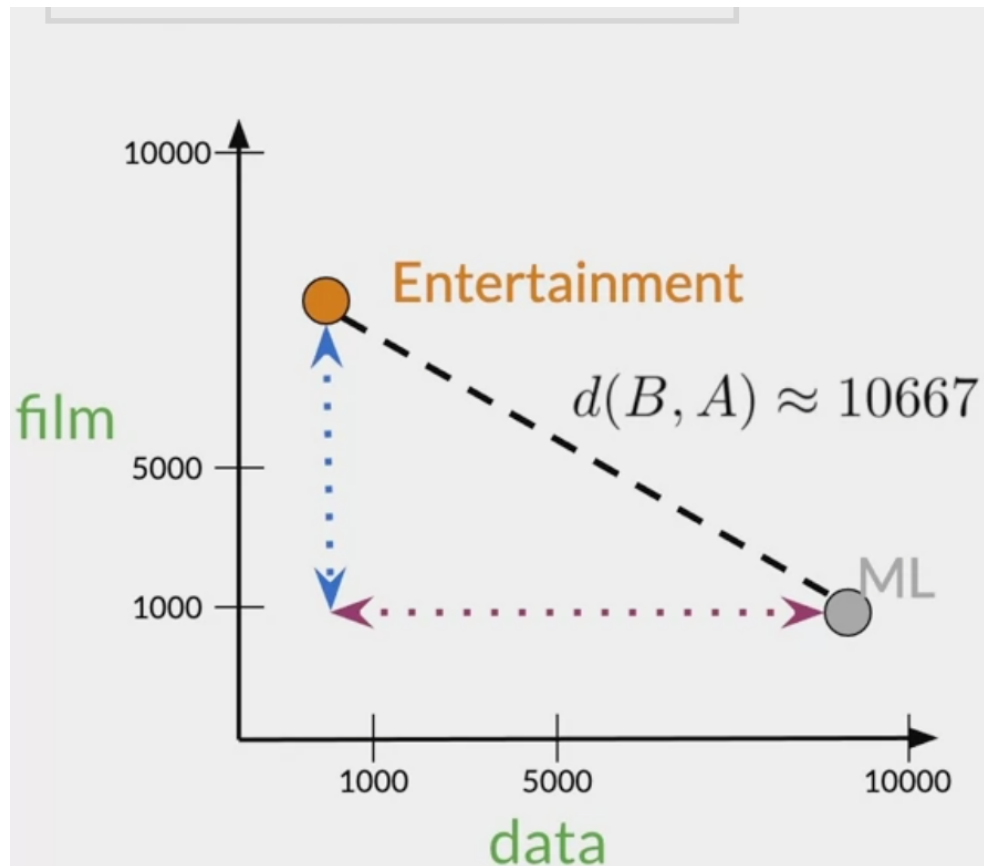
# Vector Space



# Euclidean Distance



# Euclidean Distance



Corpus **A**: (500,7000)



Corpus **B**: (9320,1000)

$$d(B, A) = \sqrt{(B_1 - A_1)^2 + (B_2 - A_2)^2}$$

$$c^2 = a^2 + b^2$$

$$d(B, A) = \sqrt{(-8820)^2 + (6000)^2}$$

# Euclidean Distance for N-Dimensional Vectors

		$\vec{w}$	$\vec{v}$
	data	boba	ice-cream
AI	6	0	1
drinks	0	4	6
food	0	6	8

$$= \sqrt{(1 - 0)^2 + (6 - 4)^2 + (8 - 6)^2}$$
$$= \sqrt{1 + 4 + 4} = \sqrt{9} = 3$$

$$d(\vec{v}, \vec{w}) = \sqrt{\sum_{i=1}^n (v_i - w_i)^2} \longrightarrow \text{Norm of } (\vec{v} - \vec{w})$$

# Euclidean Distance in Python

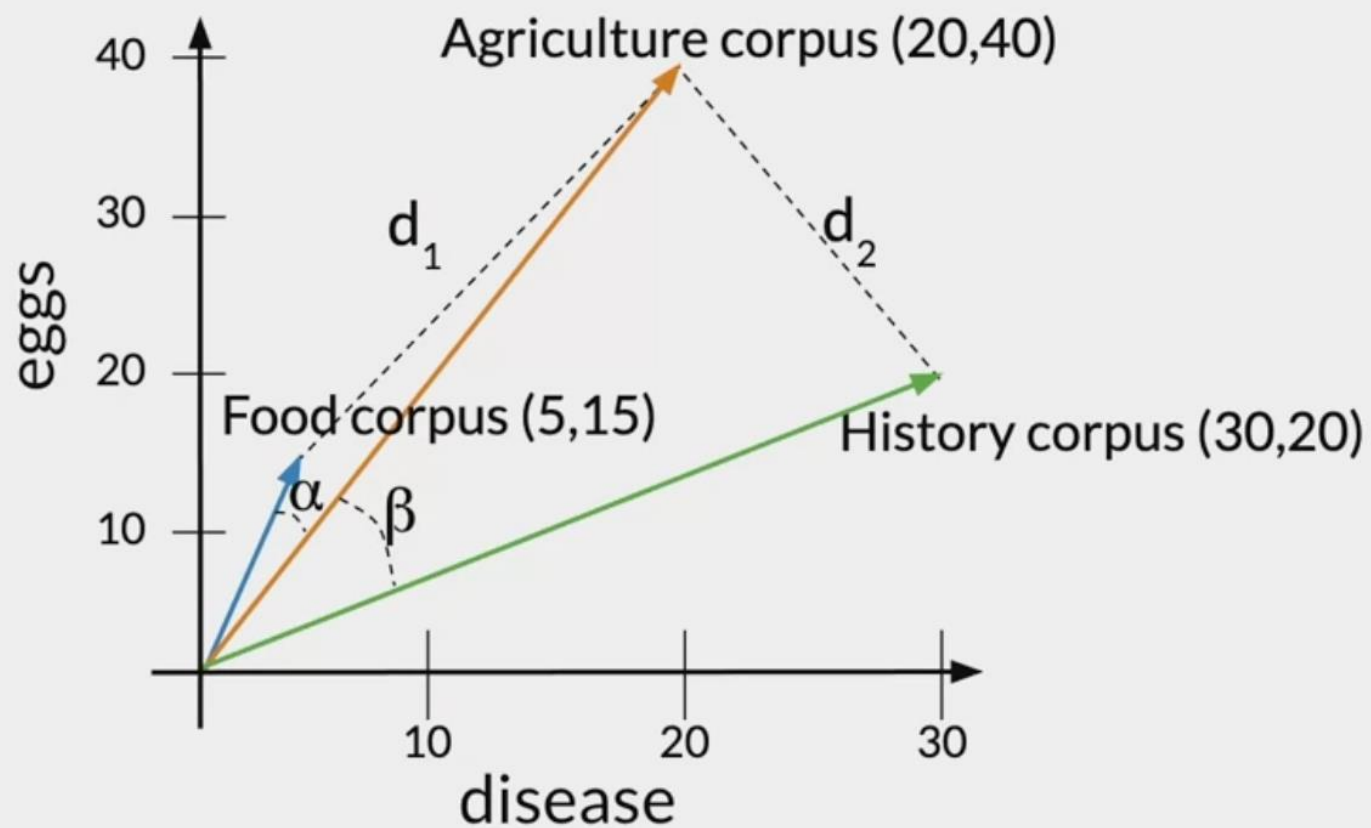
---

```
# Create numpy vectors v and w
v = np.array([1, 6, 8])
w = np.array([0, 4, 6])

# Calculate the Euclidean distance d
d = np.linalg.norm(v-w)
# Print the result
print("The Euclidean distance between v and w is: ", d)
```

The Euclidean distance between v and w is: 3

# Euclidean Distance in Python



Euclidean distance:  $d_2 < d_1$   
Angles comparison:  $\beta > \alpha$

The cosine of the angle  
between the vectors

# Previous Definition

---

Vector norm

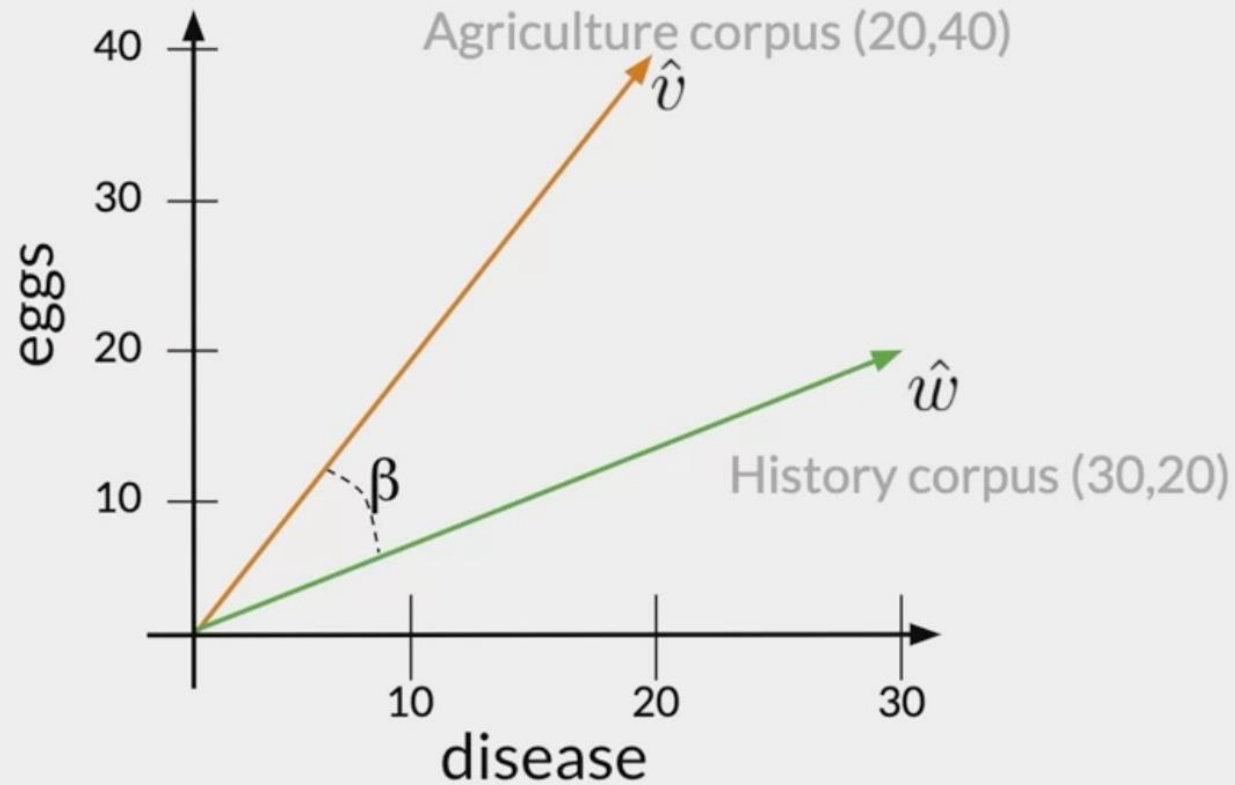
$$\|\vec{v}\| = \sqrt{\sum_{i=1}^n v_i^2}$$

Dot product

$$\vec{v} \cdot \vec{w} = \sum_{i=1}^n v_i \cdot w_i$$



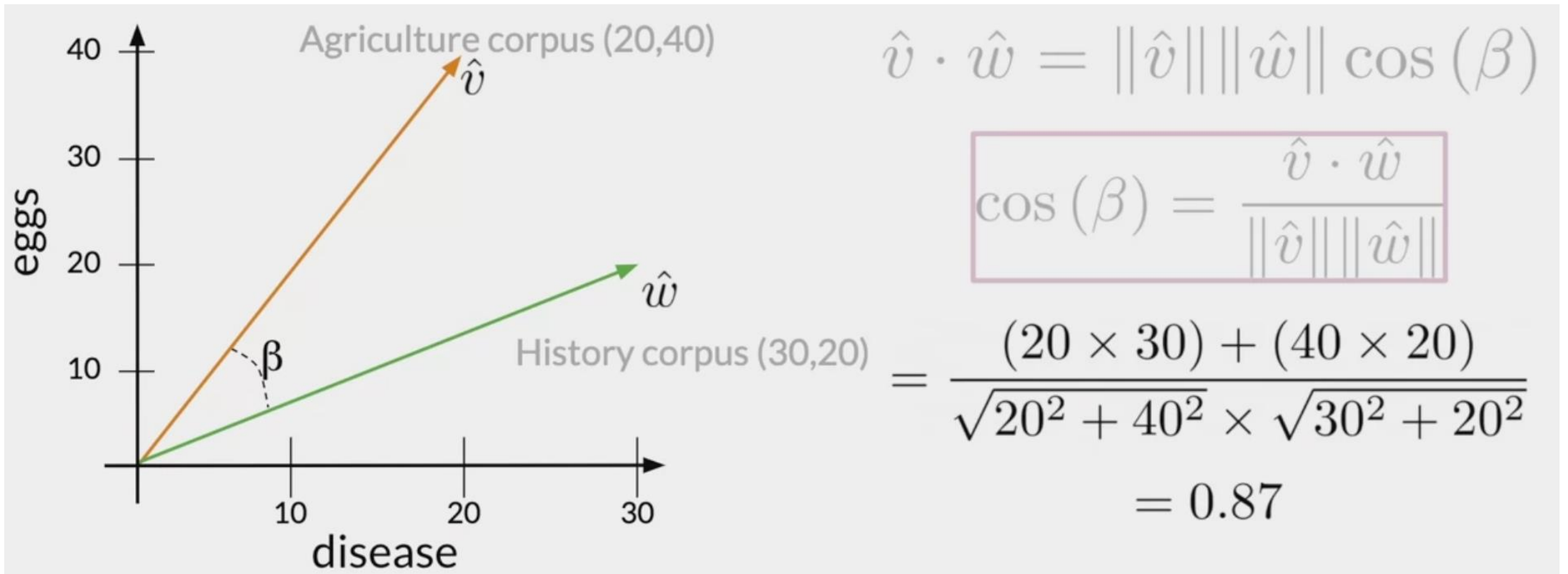
# Cosine Similarity



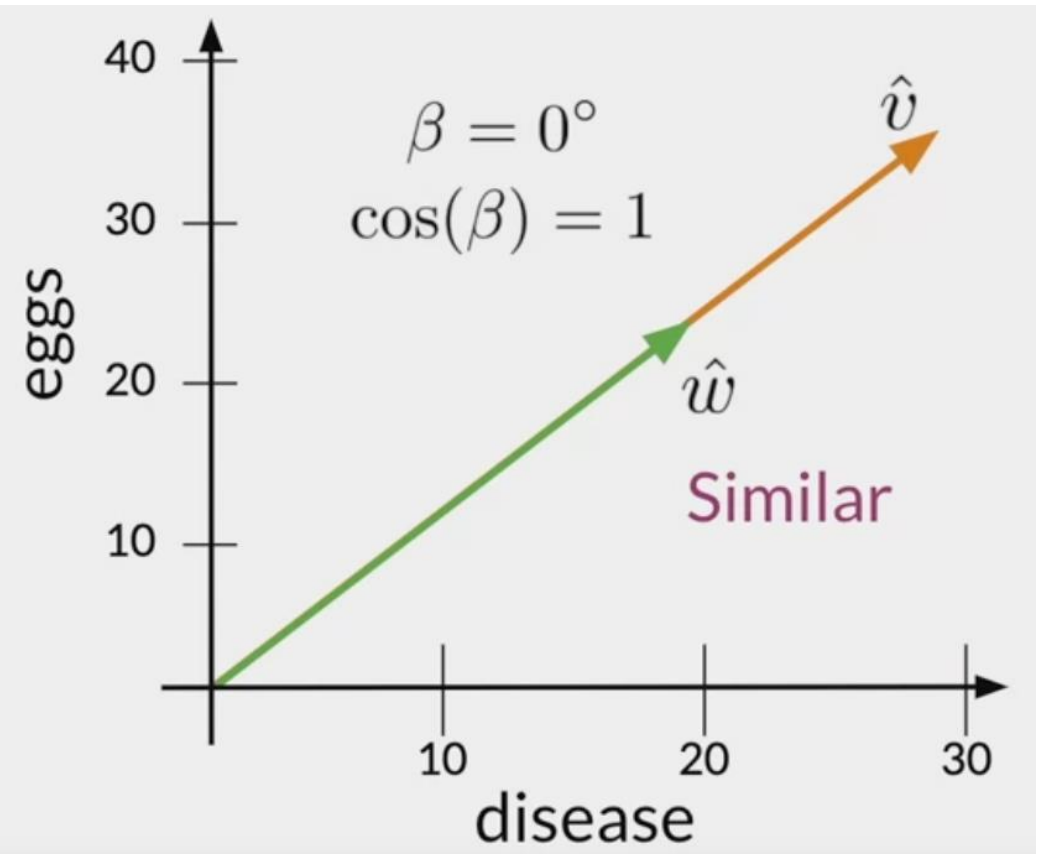
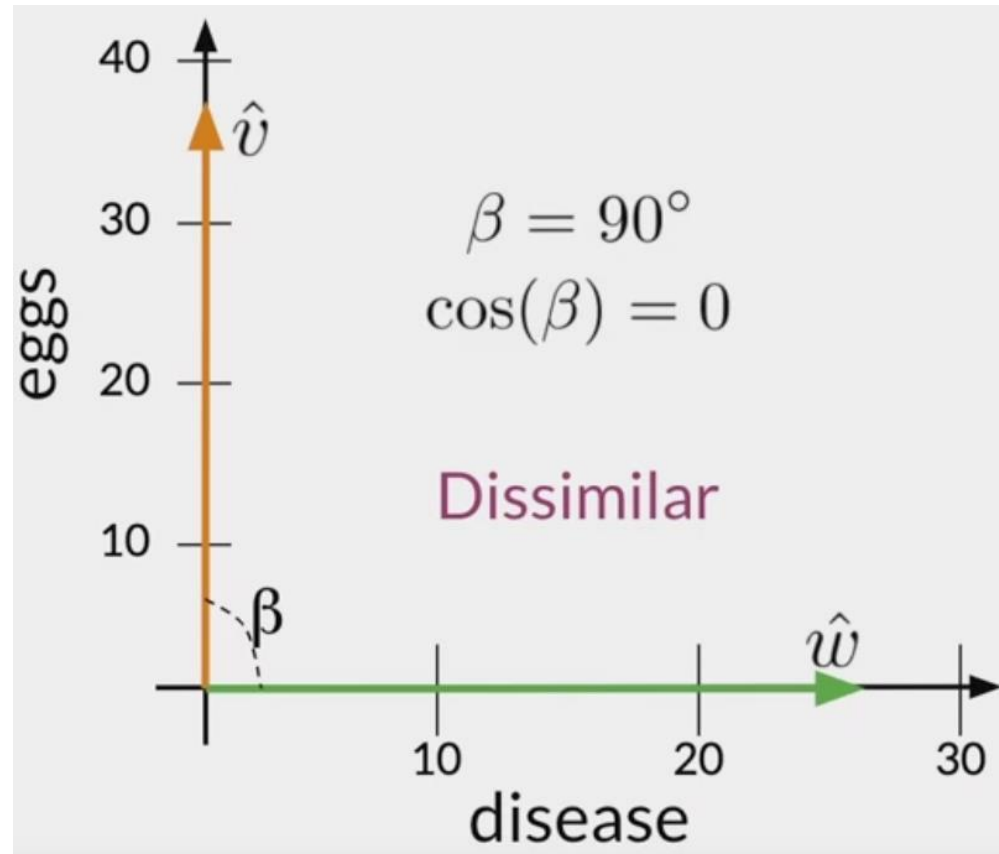
$$\hat{v} \cdot \hat{w} = \|\hat{v}\| \|\hat{w}\| \cos(\beta)$$

$$\cos(\beta) = \frac{\hat{v} \cdot \hat{w}}{\|\hat{v}\| \|\hat{w}\|}$$

# Cosine Similarity



# Cosine Similarity



# Manipulating Word Vectors



USA



Washington  
DC

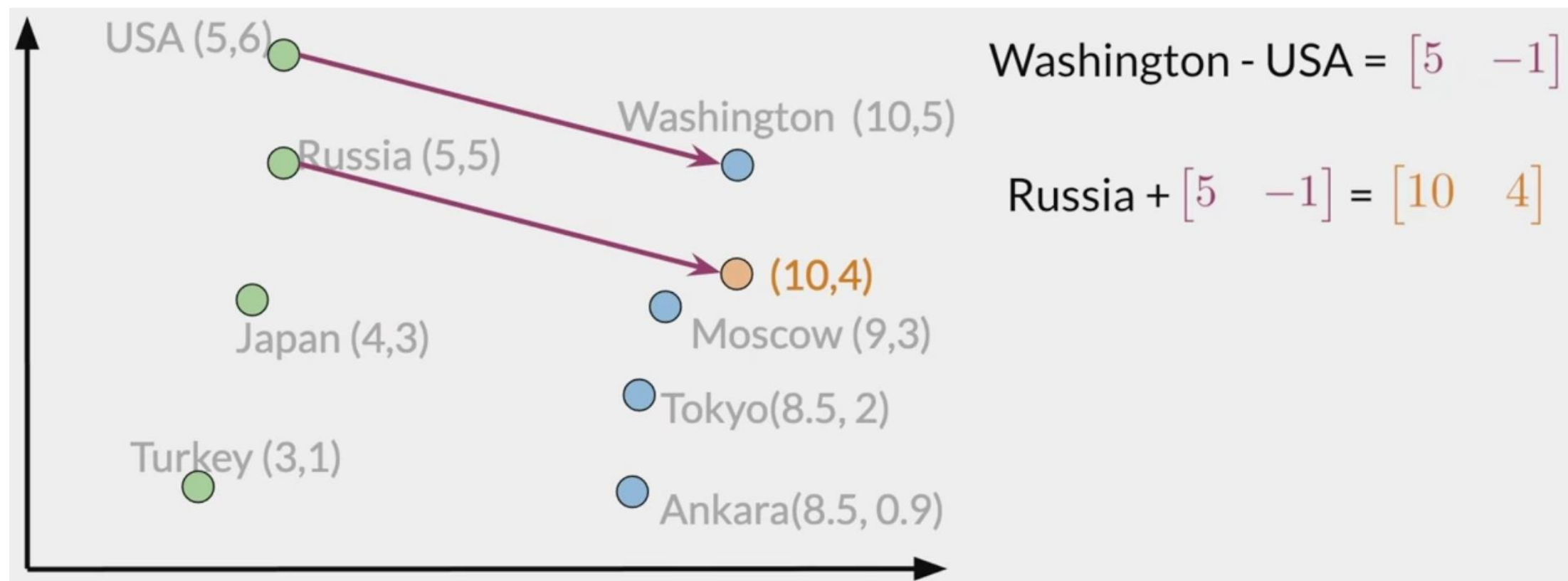


Russia

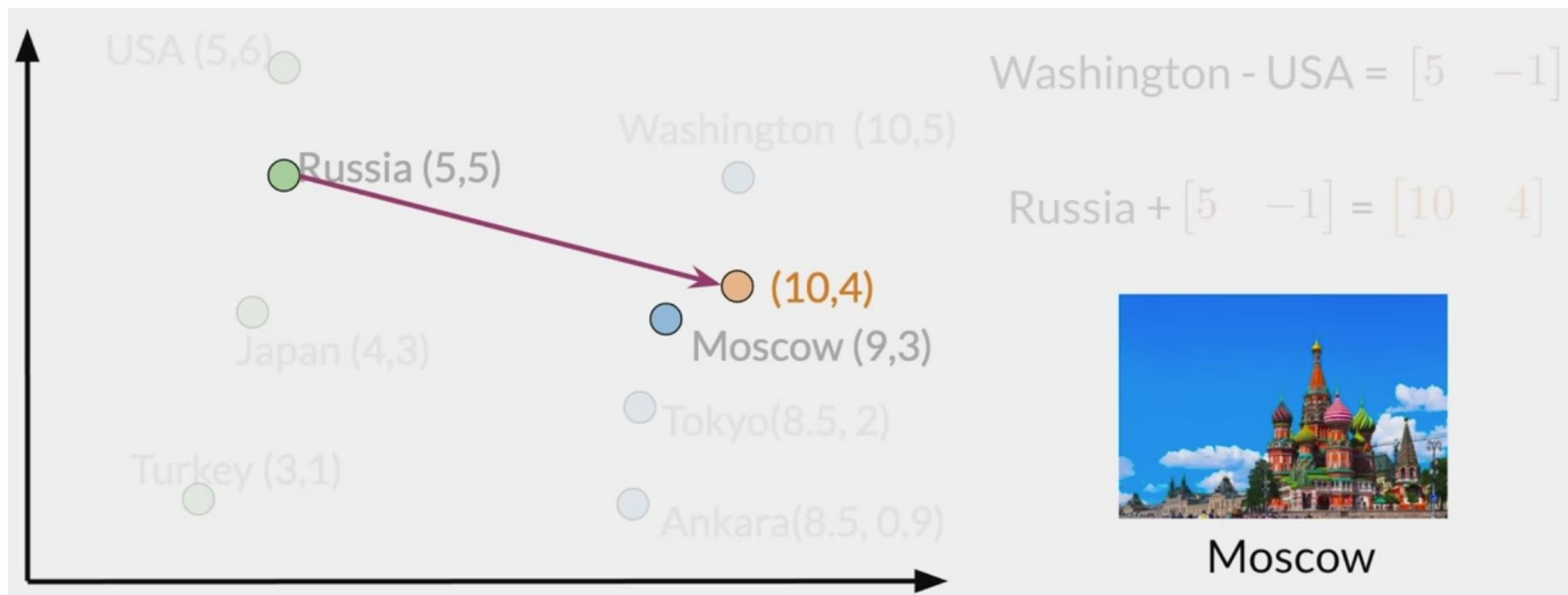


?

# Manipulating Word Vectors



# Manipulating Word Vectors



# Visualization of Word Vectors

	$d > 2$		
oil	0.20	...	0.10
gas	2.10	...	3.40
city	9.30	...	52.1
town	6.20	...	34.3

How can you visualize if your representation captures these relationships?



oil & gas



town & city

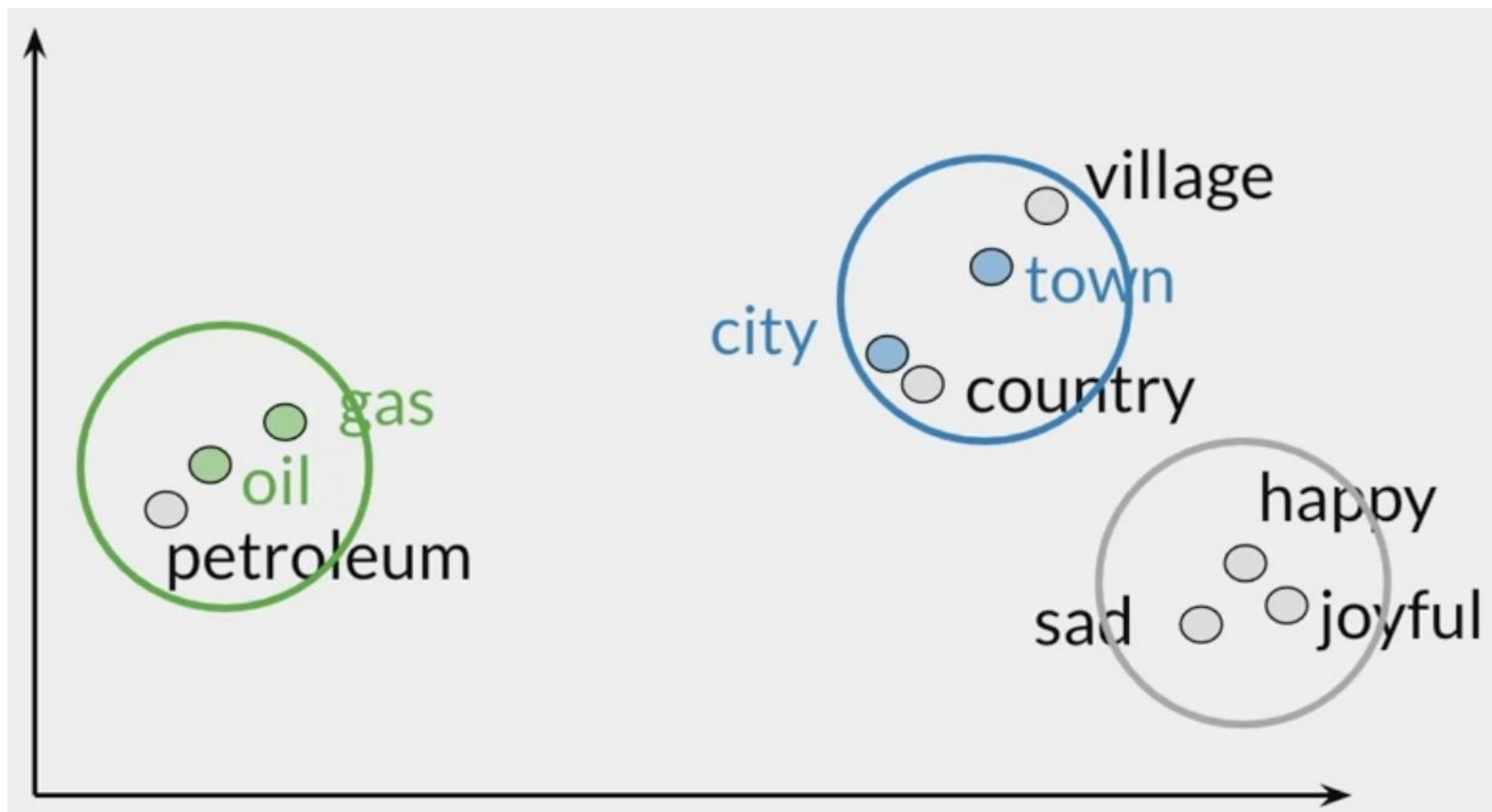
# Visualization of Word Vectors

	$d > 2$				$d = 2$	
oil	0.20	...	0.10	PCA →	oil	2.30 21.2
gas	2.10	...	3.40		gas	1.56 19.3
city	9.30	...	52.1		city	13.4 34.1
town	6.20	...	34.3		town	15.6 29.8

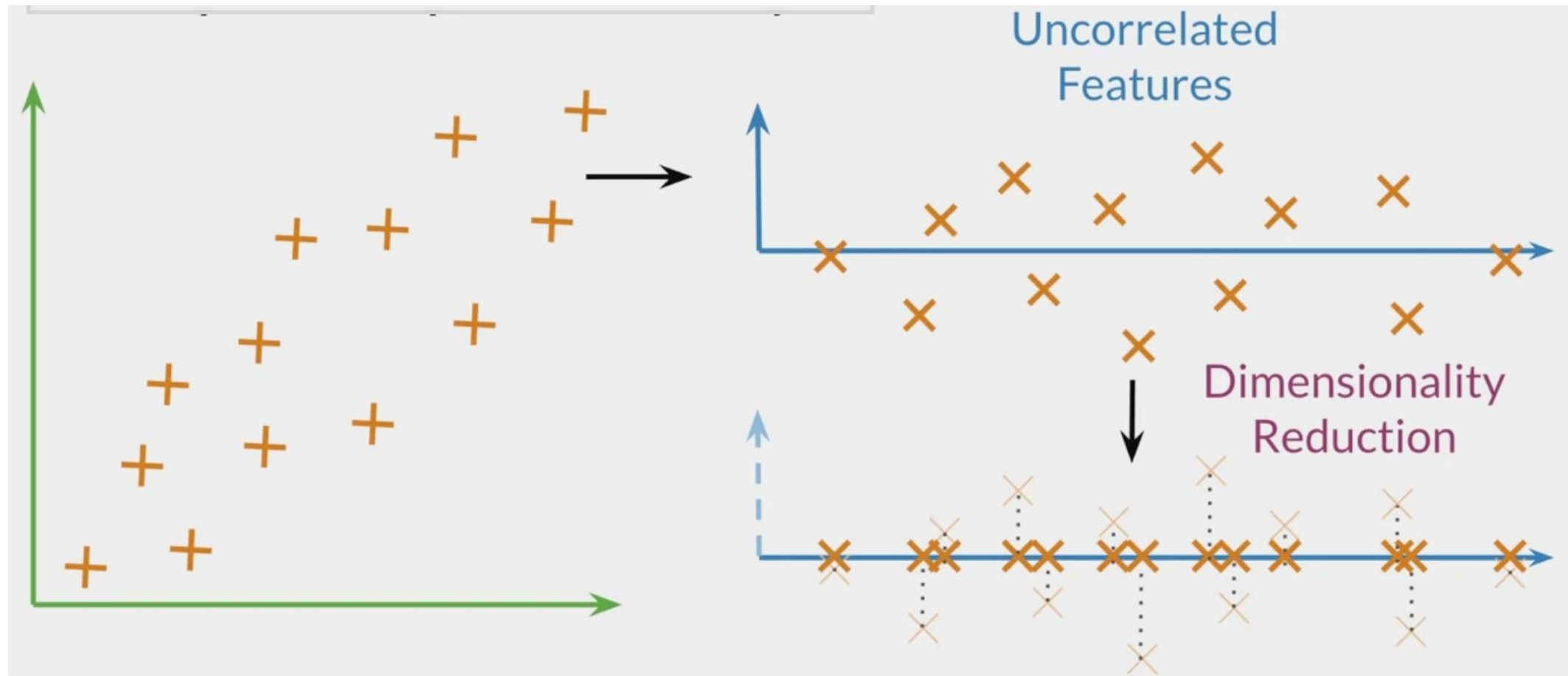


# Visualization of Word Vectors

---



# Principal Component Analysis



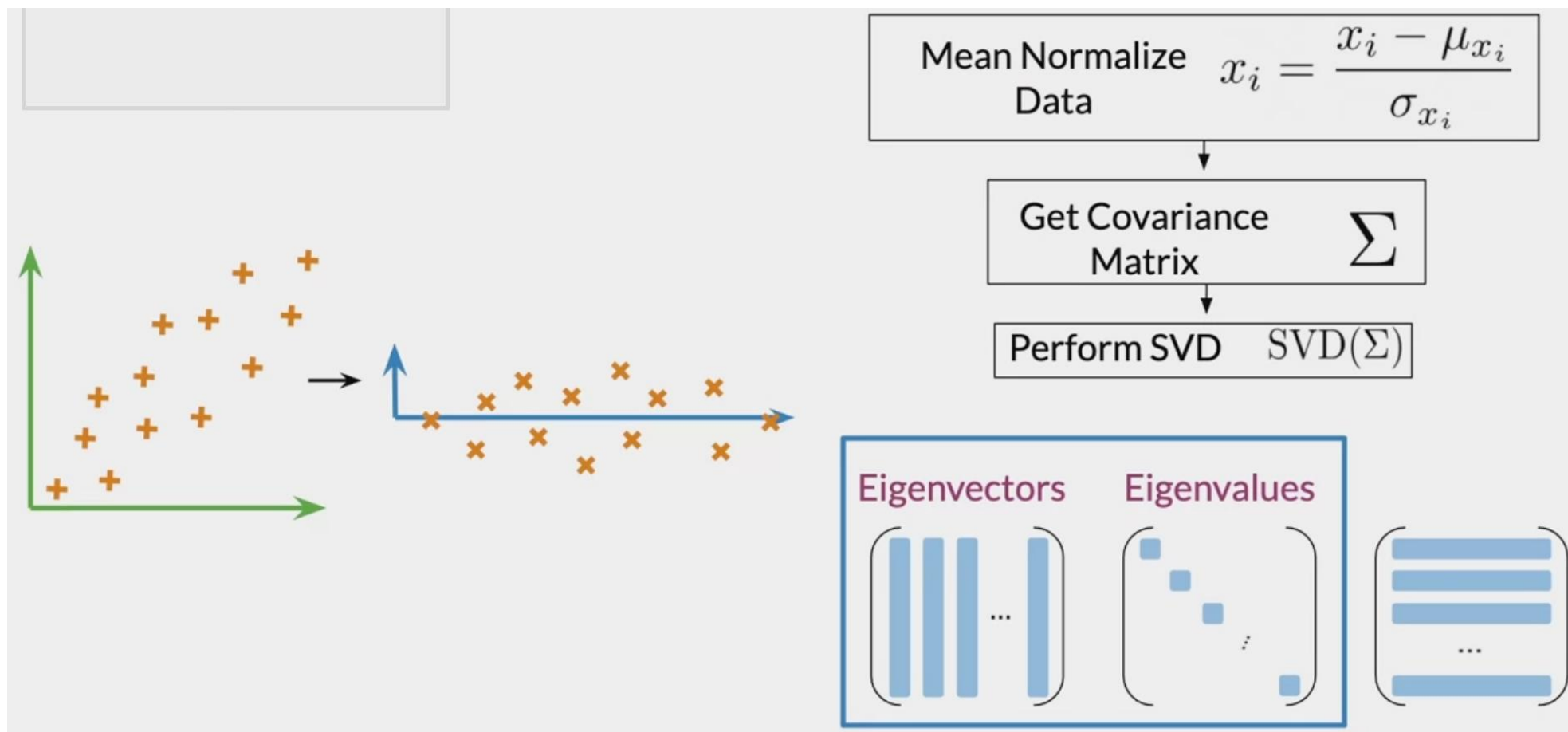
# PCA Algorithm

---

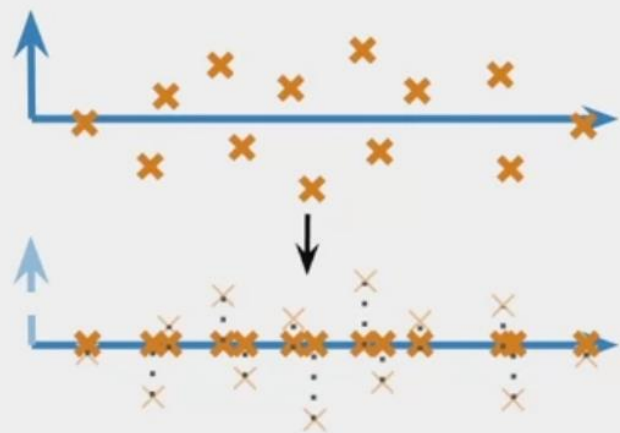
**Eigenvector:** Uncorrelated features for your data

**Eigenvalue:** the amount of information retained by each feature

# PCA Algorithm



# PCA Algorithm



Eigenvectors

$$\begin{pmatrix} | & | & | & | \\ U & & & \\ | & | & | & | \end{pmatrix}$$

Eigenvalues

$$\begin{pmatrix} \blacksquare & & & \\ & \blacksquare & & \\ & & \blacksquare & \\ & & & \blacksquare \end{pmatrix} S$$

Dot Product to  
Project Data

$$X' = XU[:, 0 : 2]$$

Percentage of  
Retained Variance

$$\frac{\sum_{i=0}^1 S_{ii}}{\sum_{j=0}^d S_{jj}}$$