DLI Accelerated Data Science Teaching Kit

Lecture 20.2 - Latent Semantic Indexing (Singular Value Decomposition)

Latent Semantic Indexing (LSI)

Main idea

- map each **document** into some '**concepts**'
- map each **term** into some '**concepts**'

'**Concept**' : ~ a set of terms, with weights.

For example, DBMS_concept:
"data" (0.8),
"system" (0.5),
"retrieval" (0.6)

Latent Semantic Indexing (LSI)

~ pictorially (before) ~

document-term matrix

	data	system	retireval	lung	ear
doc1	1	1	1		
doc2	1	1	1		
doc3				1	1
doc4				1	1

Latent Semantic Indexing (LSI)

~ pictorially (after) ~

term-concept matrix

	database concept	medical concept
data	1	
system	1	
retrieval	1	
lung		1
ear		1

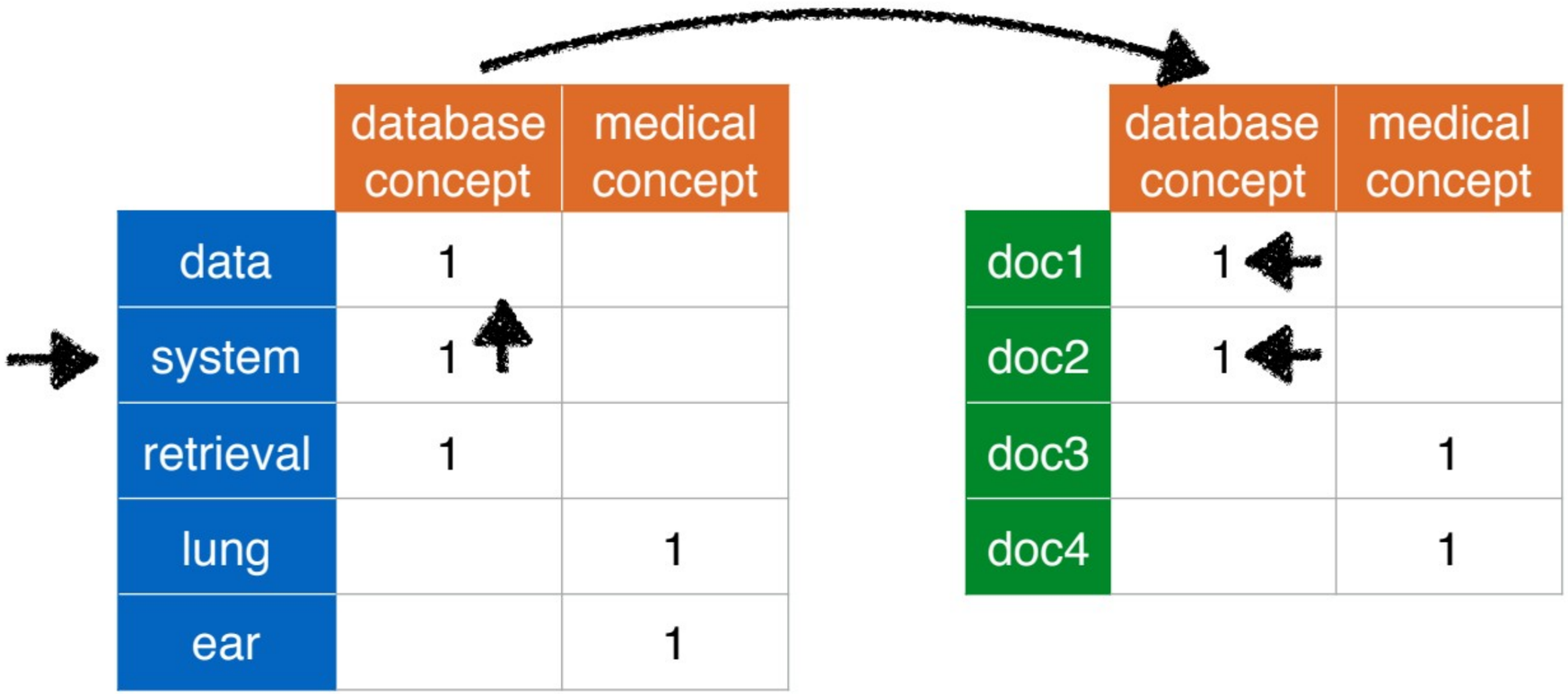
… and

document-concept matrix

	database concept	medical concept
doc1	1	
doc2	1	
doc3		1
doc4		1

Latent Semantic Indexing (LSI)

Q: How to search, e.g., for "system"?
A: find the corresponding concept(s); and the corresponding documents



	database concept	medical concept
data	1	
system	1	
retrieval	1	
lung		1
ear		1

	database concept	medical concept
doc1	1	
doc2	1	
doc3		1
doc4		1

Latent Semantic Indexing (LSI)

Works like an **automatically constructed thesaurus**

We may retrieve documents that **DON'T** have the term "system", but they contain almost everything else ("data", "retrieval")

LSI - Discussion

Great idea,

- to derive **'concepts'** from documents
- to build a **'thesaurus'** automatically
- to reduce dimensionality (down to few "concepts")

How does LSI work?

Uses **Singular Value Decomposition** (SVD)

Singular Value Decomposition (SVD)

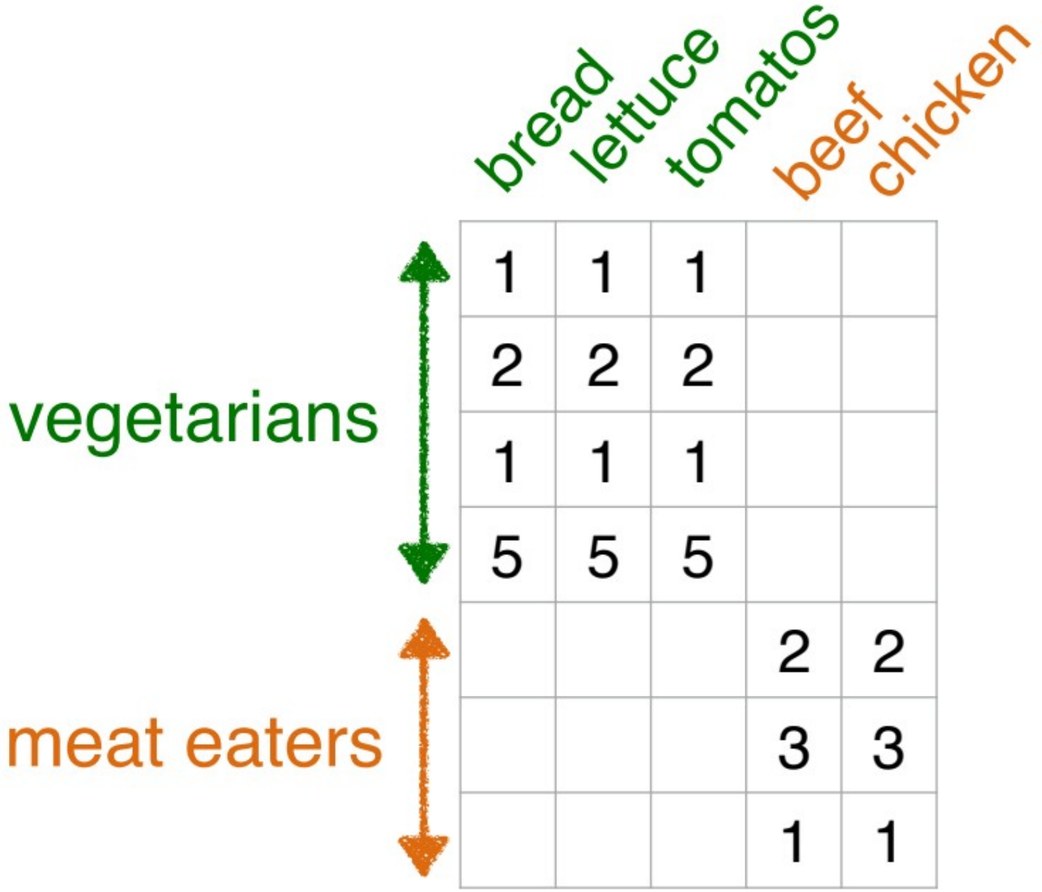
Motivation

Problem #1

Find “concepts” in matrices

Problem #2

Compression / dimensionality reduction



SVD is a powerful, generalizable technique.

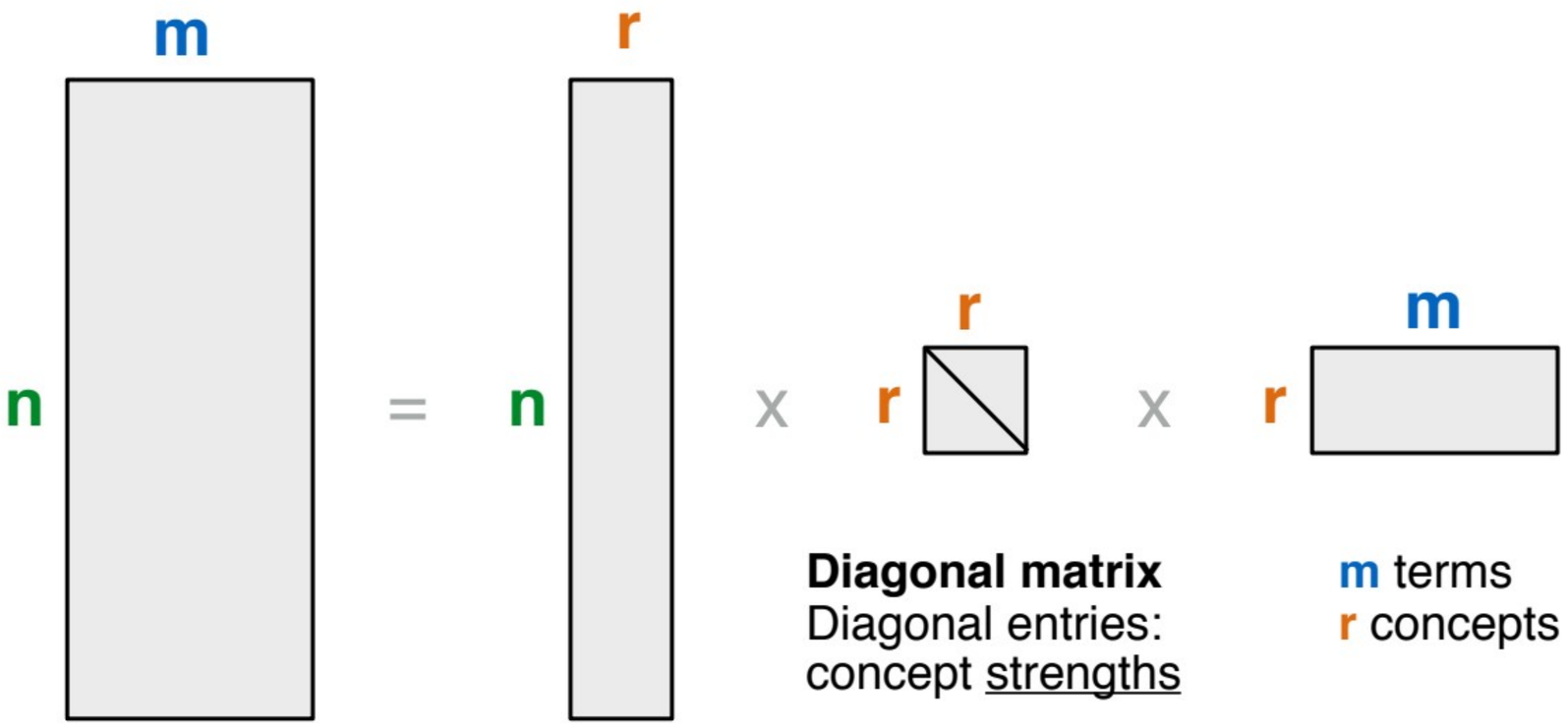
Songs / Movies / Products

Customers

1	1	1		
2	2	2		
1	1	1		
5	5	5		
			2	2
			3	3
			1	1

SVD Definition (pictorially)

$$\mathbf{A}_{[n \times m]} = \mathbf{U}_{[n \times r]} \Lambda_{[r \times r]} (\mathbf{V}_{[m \times r]})^T$$



SVD Definition (in words)

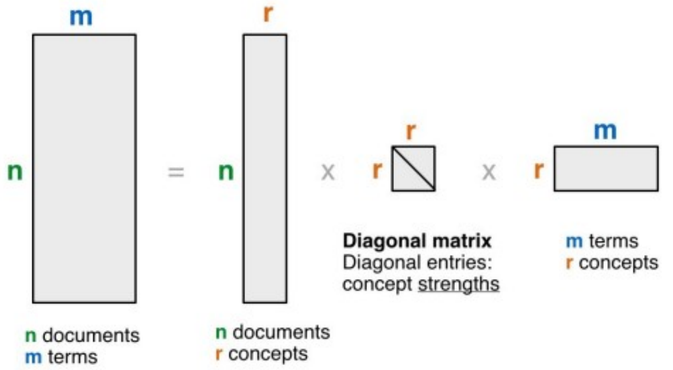
$$\mathbf{A}_{[n \times m]} = \mathbf{U}_{[n \times r]} \Lambda_{[r \times r]} (\mathbf{V}_{[m \times r]})^T$$

A: n x m matrix
e.g., n documents, m terms

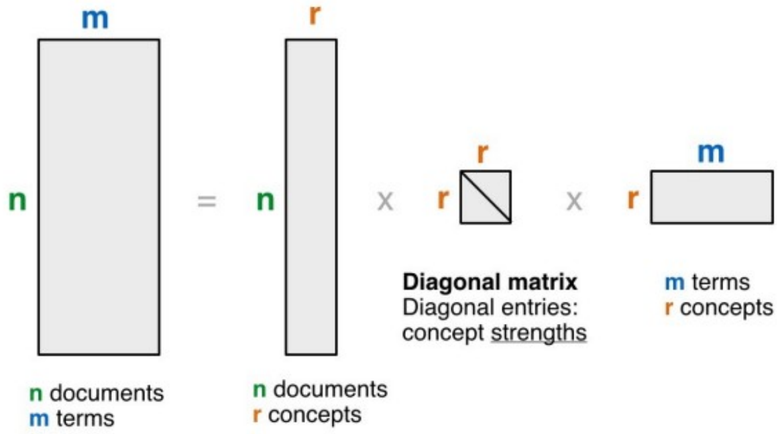
U: n x r matrix
e.g., n documents, r concepts

Λ: r x r diagonal matrix
r : rank of the matrix; strength of each 'concept'

V: m x r matrix
e.g., m terms, r concepts



SVD - Properties



THEOREM [Press+92]:

always possible to decompose matrix **A** into

$\mathbf{A} = \mathbf{U}\,\Lambda\,\mathbf{V}^T$

U, Λ, **V**: **unique**, most of the time

U, **V**: column **orthonormal**

i.e., columns are unit vectors, and orthogonal to each other

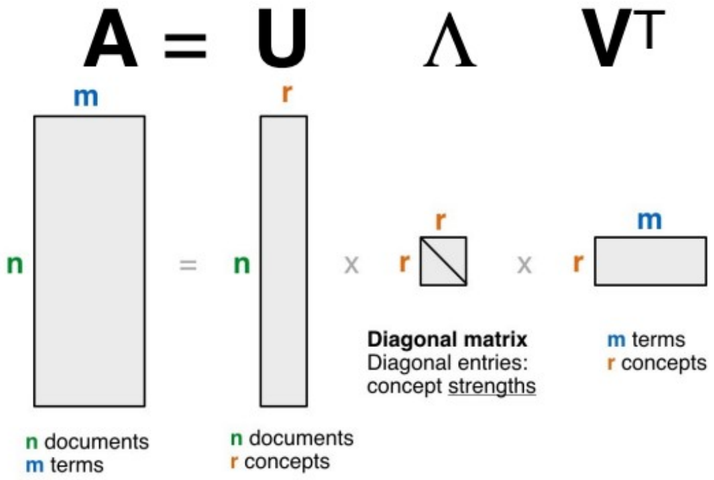
$\mathbf{U}^T\mathbf{U} = \mathbf{I}$

$\mathbf{V}^T\mathbf{V} = \mathbf{I}$

(**I:** identity matrix)

Λ: **diagonal** matrix with non-negative diagonal entires, sorted in decreasing order

SVD - Example



	data	info	retrieval	brain	lung
CS docs	1	1	1	0	0
	2	2	2	0	0
	1	1	1	0	0
	5	5	5	0	0
MD docs	0	0	0	2	2
	0	0	0	3	3
	0	0	0	1	1

=

0.18	0
0.36	0
0.18	0
0.90	0
0	0.53
0	0.80
0	0.27

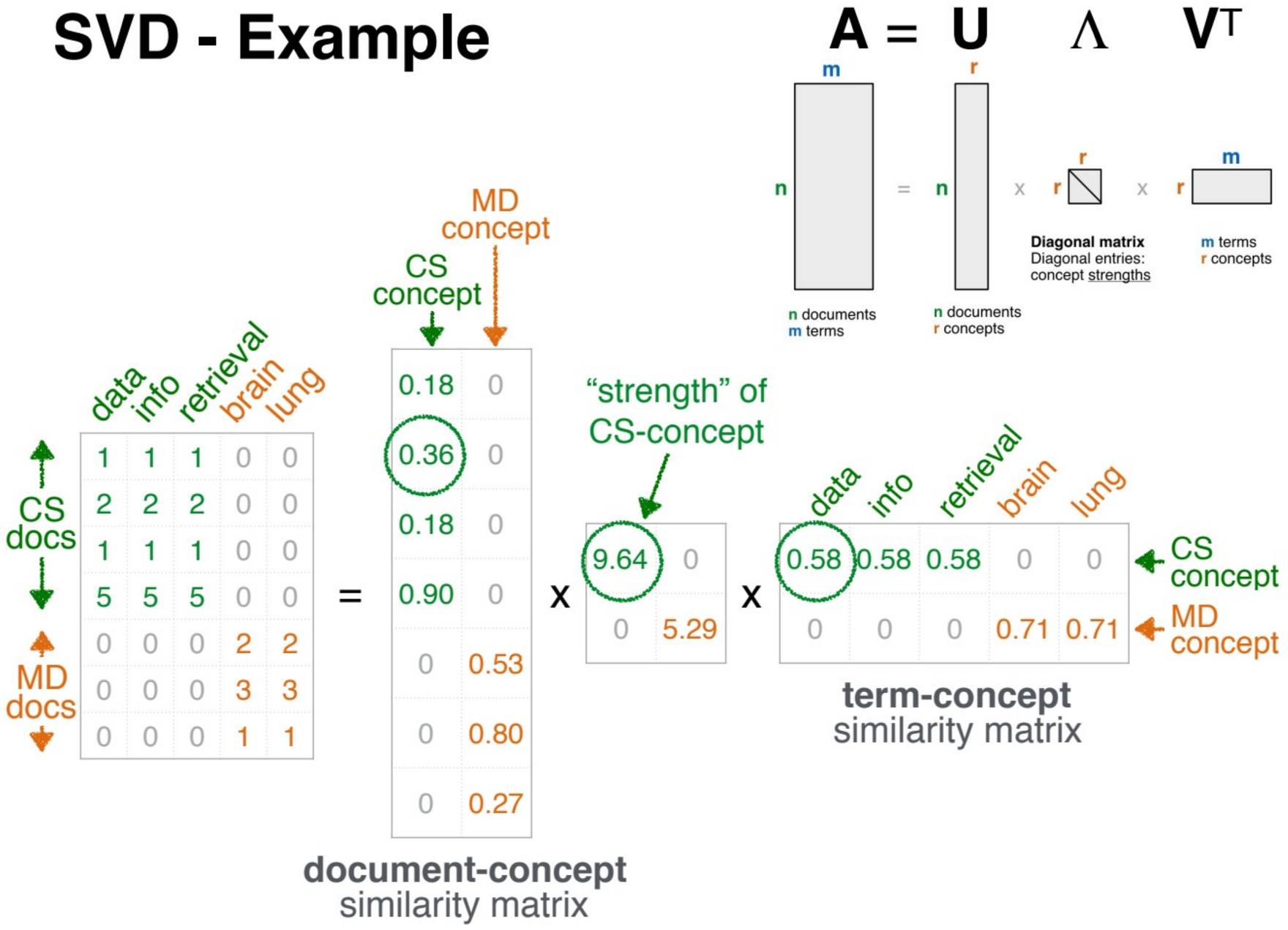
x

9.64	0
0	5.29

x

0.58	0.58	0.58	0	0
0	0	0	0.71	0.71

SVD - Example



DLI Accelerated Data Science Teaching Kit

Thank You