

## Netflix intro

What data should we encode about each Netflix account holder to help us make effective recommendations?

In machine learning, clustering can be used to group similar data for prediction and recommendation. For example, each Netflix user's viewing history can be represented as a  $n$ -tuple indicating their preferences about movies in the database, where  $n$  is the number of movies in the database. People with similar tastes in movies can then be clustered to provide recommendations of movies for one another. Mathematically, clustering is based on a notion of distance between pairs of  $n$ -tuples.

# Set operations

To define a set we can use the roster method, set builder notation, a recursive definition, and also we can apply a set operation to other sets.

**New! Cartesian product of sets and set-wise concatenation of sets of strings**

**Definition:** Let  $A$  and  $B$  be sets. The **Cartesian product** of  $A$  and  $B$ , denoted  $A \times B$ , is the set of all ordered pairs  $(a, b)$  where  $a \in A$  and  $b \in B$

$$A \times B = \{(a, b) \mid a \in A \text{ and } b \in B\}$$

**Definition:** Let  $A$  and  $B$  be sets of strings over the same alphabet. The **set-wise concatenation** of  $A$  and  $B$ , denoted  $A \circ B$ , is the set of all results of string concatenation  $ab$  where  $a \in A$  and  $b \in B$

$$A \circ B = \{ab \mid a \in A \text{ and } b \in B\}$$

Fill in the missing entries in the table:

Set	Example elements in this set:			
$B$	A	C	G	U
	(A, C)		(U, U)	
$B \times \{-1, 0, 1\}$				
$\{-1, 0, 1\} \times B$				
	(0, 0, 0)			
$\{A, C, G, U\} \circ \{A, C, G, U\}$				
	GGGG			

# Defining functions

**New! Defining functions** A function is defined by its (1) domain, (2) codomain, and (3) rule assigning each element in the domain exactly one element in the codomain.

The domain and codomain are nonempty sets.

The rule can be depicted as a table, formula, or English description.

Example: The absolute value function

**Domain**

**Codomain**

**Rule**

In the table below, each row represents a user's ratings of movies: ✓ (check) indicates the person liked the movie, ✗ (x) that they didn't, and • (dot) that they didn't rate it one way or another (neutral rating or didn't watch).

Person	Fyre	Frozen II	Picard	Ratings written as a 3-tuple
$P_1$	✗	•	✓	$(-1, 0, 1)$
$P_2$	✓	✓	✗	$(1, 1, -1)$
$P_3$	✓	✓	✓	$(1, 1, 1)$
$P_4$	•	✗	✓	

Which of  $P_1$ ,  $P_2$ ,  $P_3$  has movie preferences most similar to  $P_4$ ?

One approach to answer this question: use **functions** to define distance between user preferences.

Define the following functions whose inputs are ordered pairs of 3-tuples each of whose components comes from the set  $\{-1, 0, 1\}$

$$d_1((x_1, x_2, x_3), (y_1, y_2, y_3)) = \sum_{i=1}^3 ((|x_i - y_i| + 1) \mathbf{div} 2)$$

$$d_2((x_1, x_2, x_3), (y_1, y_2, y_3)) = \sqrt{\sum_{i=1}^3 (x_i - y_i)^2}$$

$d_1(P_4, P_1)$	$d_1(P_4, P_2)$	$d_1(P_4, P_3)$
$d_2(P_4, P_1)$	$d_2(P_4, P_2)$	$d_2(P_4, P_3)$

*Extra example:* A new movie is released, and  $P_1$  and  $P_2$  watch it before  $P_3$ , and give it ratings;  $P_1$  gives ✓ and  $P_2$  gives ✗. Should this movie be recommended to  $P_3$ ? Why or why not?

*Extra example:* Define the new functions that would be used to compare the 4-tuples of ratings encoding movie preferences now that there are four movies in the database.

## Defining functions recursively

**Definition** (Of a function, recursively) A function *rnalen* that computes the length of RNA strands in  $S$  is defined by:

$$\begin{array}{lll}
 & & \text{rnalen} : S \rightarrow \mathbb{Z}^+ \\
 \text{Basis Step:} & \text{If } b \in B \text{ then} & \text{rnalen}(b) = 1 \\
 \text{Recursive Step:} & \text{If } s \in S \text{ and } b \in B, \text{ then} & \text{rnalen}(sb) = 1 + \text{rnalen}(s)
 \end{array}$$

The domain of *rnalen* is \_\_\_\_\_. The codomain of *rnalen* is \_\_\_\_\_.

$$\text{rnalen}(\text{ACU}) = \underline{\hspace{10cm}}$$

*Extra example:* A function *basecount* that computes the number of a given base  $b$  appearing in a RNA strand  $s$  is defined recursively: *fill in codomain and sample function applications*

$$\begin{array}{lll}
 & & \text{basecount} : S \times B \rightarrow \\
 \text{Basis Step:} & \text{If } b_1 \in B, b_2 \in B & \text{basecount}(b_1, b_2) = \begin{cases} 1 & \text{when } b_1 = b_2 \\ 0 & \text{when } b_1 \neq b_2 \end{cases} \\
 \text{Recursive Step:} & \text{If } s \in S, b_1 \in B, b_2 \in B & \text{basecount}(sb_1, b_2) = \begin{cases} 1 + \text{basecount}(s, b_2) & \text{when } b_1 = b_2 \\ \text{basecount}(s, b_2) & \text{when } b_1 \neq b_2 \end{cases}
 \end{array}$$

$$\text{basecount}(\text{ACU}, \text{A}) = \underline{\hspace{10cm}}$$

$$\text{basecount}(\text{ACU}, \text{G}) = \underline{\hspace{10cm}}$$