

## Ratings examples

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 $\{-1, 0, 1\}$	
$d_1((x_1, x_2, x_3), (y_1, y_2, y_3)) = \sum_{i=1}^3 (( x_i - y_i  + 1) \text{ div } 2)$	$d_2((x_1, x_2, x_3), (y_1, y_2, y_3)) = \sqrt{\sum_{i=1}^3 (x_i - y_i)^2}$

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

*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.