Who likes the same TV shows?

Ursula, Victoria, Wendell, and Xavier are asked to rate TV shows *Uptown Abbey*, *Game of Scones*, and *The Real Housewives of Milpitas* on a scale from -4 (hate it) to $+4$ (love it). Their responses:

$$\begin{aligned}\mathbf{u} &= \langle -2, -1, 4 \rangle\,, \\ \mathbf{v} &= \langle 2, 4, -2 \rangle\,, \\ \mathbf{w} &= \langle 1, 4, -1 \rangle\,, \\ \mathbf{x} &= \langle 0, -3, -4 \rangle\,.\end{aligned}$$

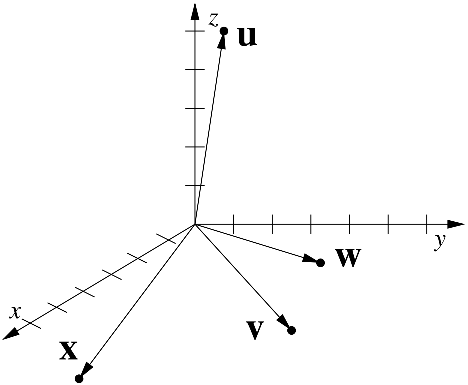
Whose tastes in TV are the most similar? Whose are the most different?

Graphing the opinion vectors

Below, we see that it is reasonable to think of two vectors as similar if

- they form a small angle (i.e., $\cos\theta \approx 1$) or
- they form an acute angle and the vectors themselves are long.

The latter means that passionate fans whose tastes might sometimes differ have a high similarity score.



The solution: dot products!

Putting those two ideas together, we see that two vectors represent similar tastes if their *dot products* are large and positive.
In our example, we have

$$\mathbf{u}\cdot\mathbf{v} = -16, \qquad \mathbf{u}\cdot\mathbf{w} = -10, \qquad \mathbf{u}\cdot\mathbf{x} = -13,$$
$$\mathbf{v}\cdot\mathbf{w} = 20, \qquad \mathbf{v}\cdot\mathbf{x} = -4, \qquad \mathbf{w}\cdot\mathbf{x} = -8.$$

So Victoria and Wendell have the most similar taste in TV, and Ursula and Victoria have the most different (diametrically opposed).
If, for example, you ran a streaming TV company, you could use this idea to generate suggestions for a customer to view.
("Wendell liked the movie, so you might like it!")
Of course, since the dot product works in n dimensions, you would probably use $\mathbb{R}^{30}$ (opinions on 30 shows) or $\mathbb{R}^{300}$ instead of $\mathbb{R}^3$.

Machine learning: kernel methods

The super-fancy version of the dot product idea is called the *kernel trick*.

- Conceptually, the idea is that we map (project[1]) our data space $\mathbb{R}^3$ in a very nonlinear way into some super-high-dimensional space (think $\mathbb{R}^\infty$!!) in which the pattern we seek to understand can be seen in terms of the dot product.
- In practice, a mathematical trick allows us to compute dot products directly, based on lots of known samples, or *training data*, without having to write down the (infinitely complicated) infinite-dimensional map (projection).

See the Wikipedia entry on kernel methods.

[1] Apology: Using the word "projection" here is mathematically incorrect but gives the right metaphorical idea.