Exploring Vector Embeddings and Neural Networks

Wednesday 1st October

Maciej Miazek

Projecting into n dimensions

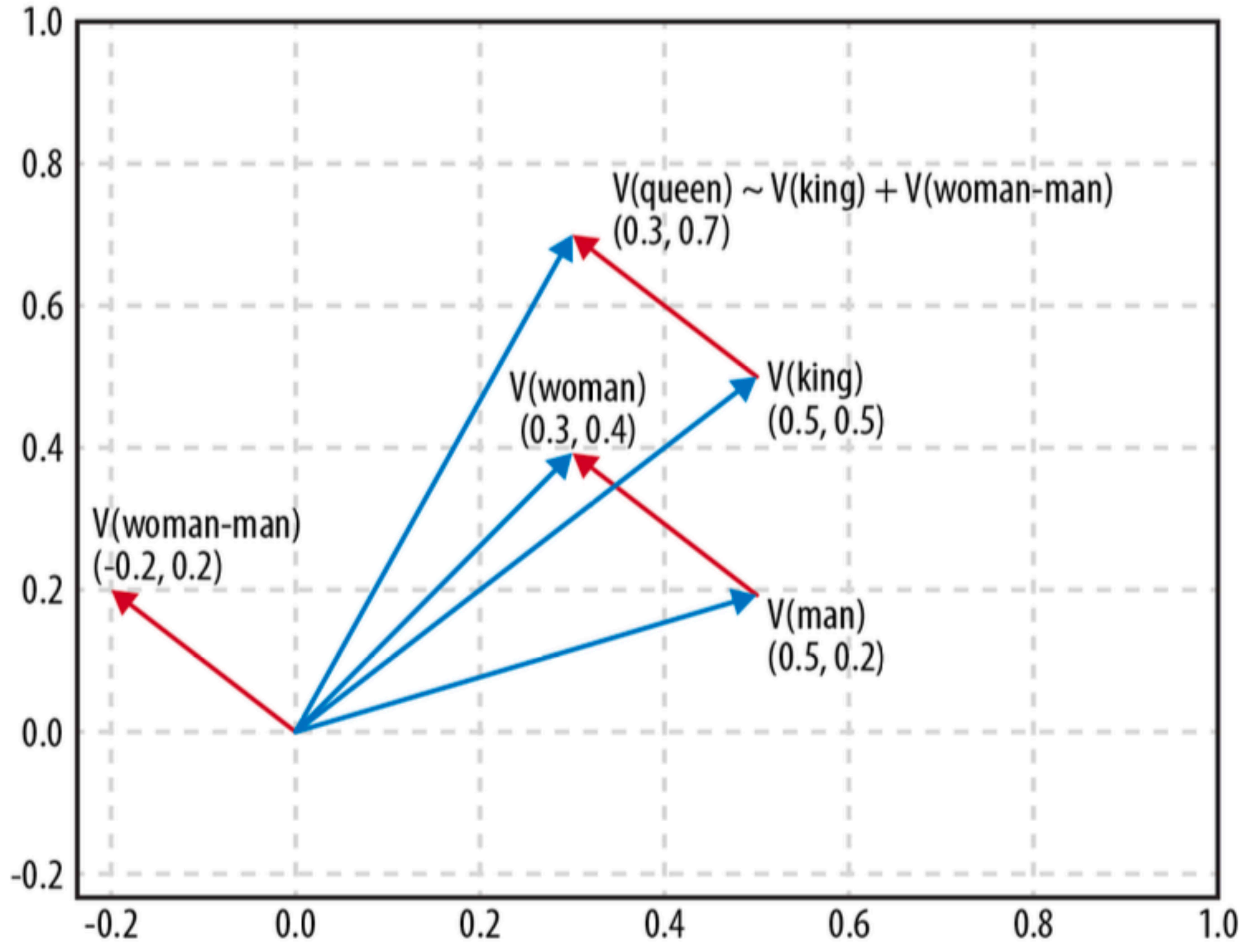
Vector Embeddings

- Word embeddings associate an n-dimensional vector with each word in the vocabulary in such a way that similar words are next to each other (semantic clustering)
- “The cafe served a {X} that really woke me up”
- Word2Vec is a simple “lookup table” rather than a deep nn but they will still let us explore the maths that comes with vector embeddings
- The Word2Vec embeddings are a side effect of training a network to predict a word from context for sentences taken from Google News (This is how you can extract semantic meaning!)

Clustering by semantic meaning

Vector maths and SVMs

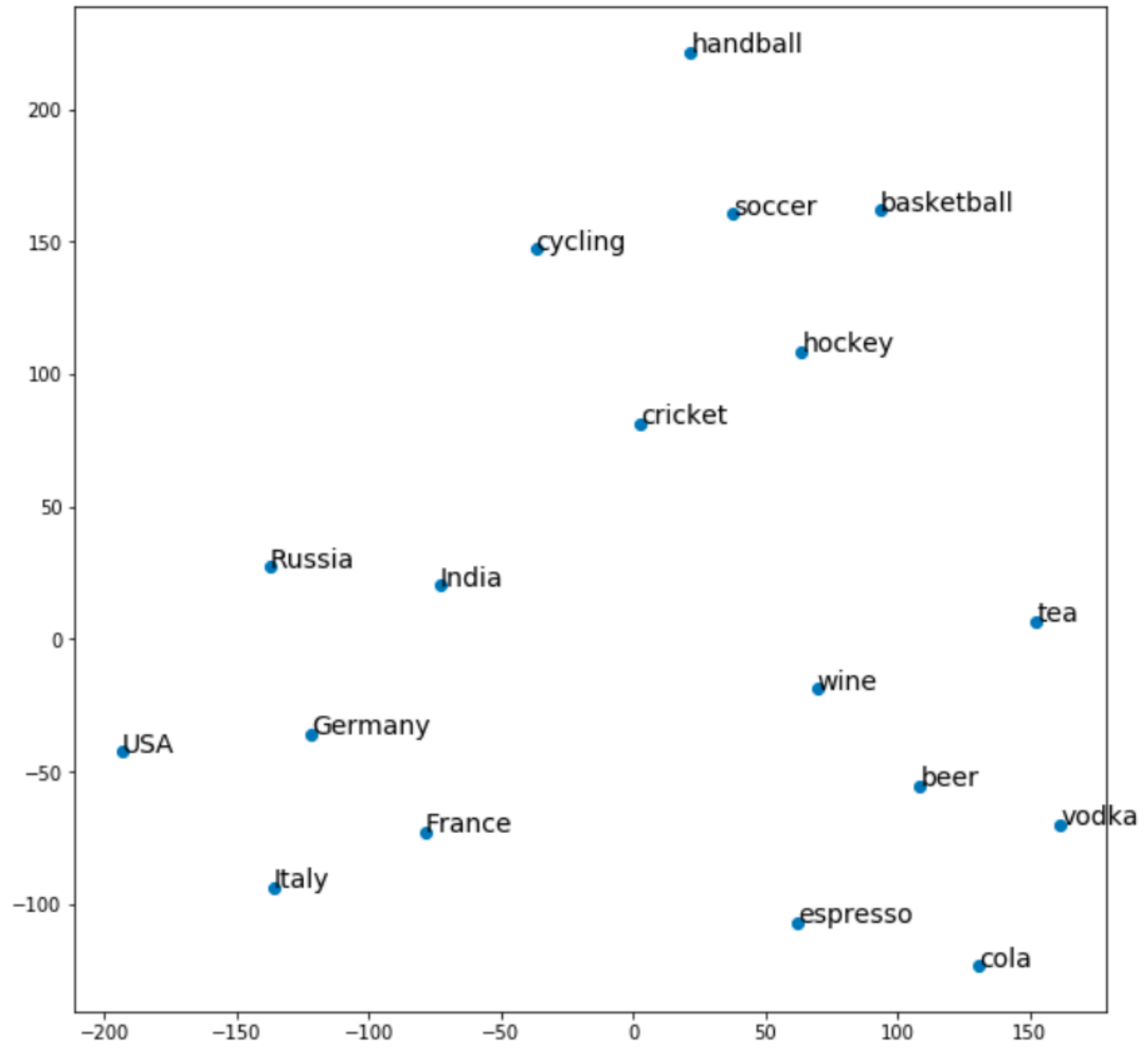
- Think back to secondary school maths/ physics to what a vector (representation of direction and magnitude of force in physics)
- This intuition carries on to however many dimensions you want to work in
- *<u>directions represent concepts in an n-dimensional space</u>*
- A_is_to_B_as_C_is_to() => B-A+C = D



How do we project n-dimensions onto 2?

T-distributed stochastic neighbour embedding (t-SNE)

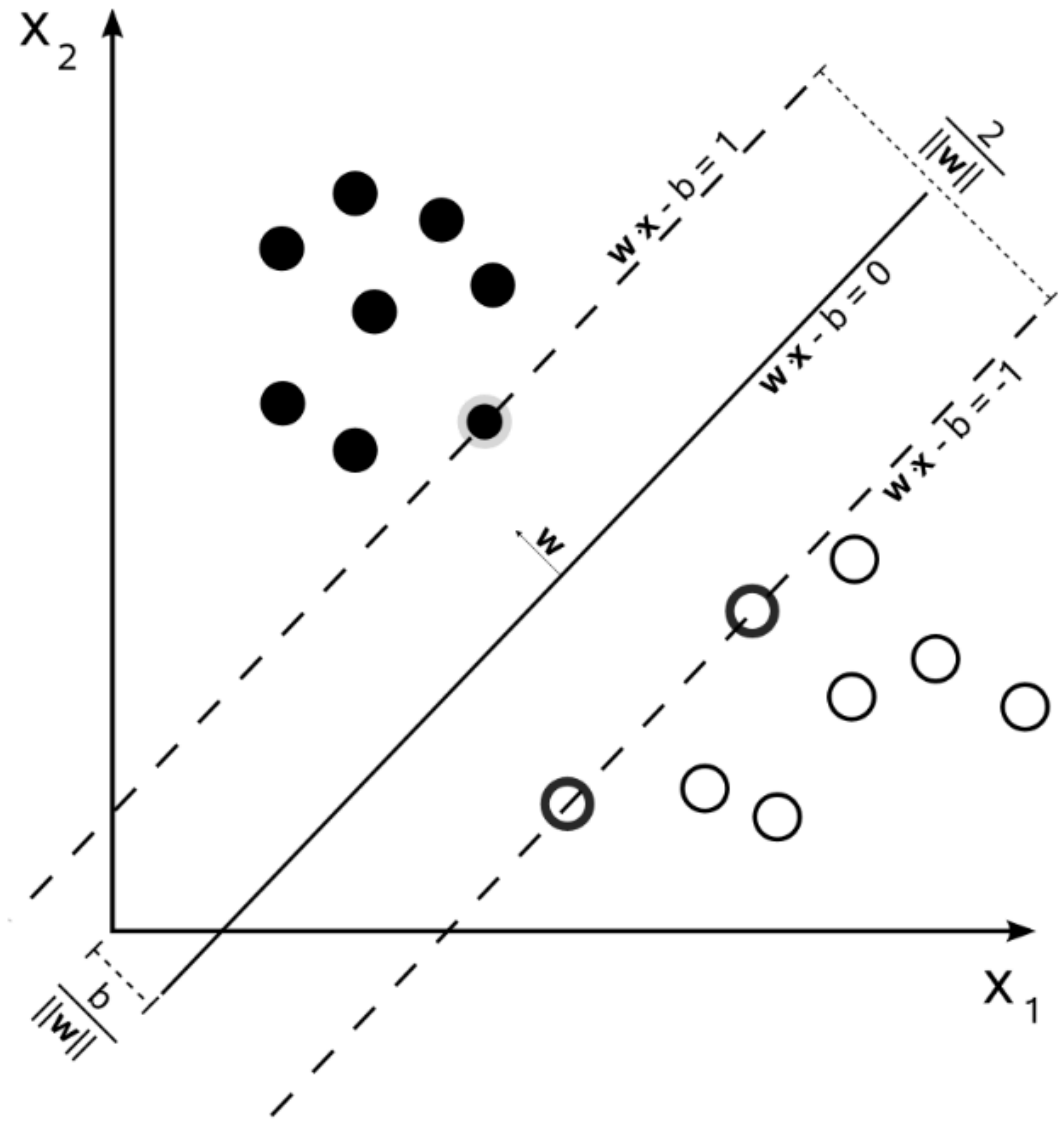
- t-SNE iteratively tries to find the best projection onto a lower-dimensional space that maintains the distances between the points as well as possible
- Given 3 arrays of 300 dimensional embeddings grouped into categories we get this ->



How does Netflix know what to recommend you?

The Support Vector Machine (SVM)

- Let's take "a country" as our example concept
- The concept of a country in the embedding space is not a point <u>but a shape</u>
- What if we prefer one shape/set over another?
- The SVM tries to find a hyperplane that separates what we prefer to what we dislike.


x_2
x_1
$\mathbf{w}\cdot\mathbf{x}-b=1$
$\mathbf{w}\cdot\mathbf{x}-b=0$
$\mathbf{w}\cdot\mathbf{x}-b=-1$
$\frac{2}{\|\mathbf{w}\|}$
$\frac{b}{\|\mathbf{w}\|}$
$\mathbf{w}$

Let’s train our own recommender

Semantically clustering vector embeddings

- We treat outgoing wikipedia links as connectors.
- The intuition here is that movies linking to the same page are similar (same director, same genre, actors)
- For example the movie “No Country for Old Men” will link to the Coen brothers, thrillers, movies made in 2007 etc.
- We take the link_id and movie_id, get a vector, then set the dot product (multiply them together) of the two as the output of the model.
- We are asking to model to come up with a space such that vector for movie and link can be used to predict whether or not they will co-occur

So now the movies are semantically clustered

Apply the SVM!

- We give our trained model an array of movies we like and movies we dislike.
- We run the SVM classifier over all the movies in our dataset
- The SVM will try to separate the two shapes
- Print the best and worst 5
- The movies that are furtherest from the separating hyperplane are movies that are either liked best or worst (depending on which side)

Quick RNN showcase

Since I promised

- A Recurrent Neural Network (RNN) paired with Long-Short term memory (LSTM) can predict sequences of tokens (either words or characters).
- LSTM allows for capturing of patterns in long sequences (transformer attention block improve on this) by letting the neural network decide which pattern is more important
- For example in programming there is a lot of sequence repetition due to the pre-set syntax. e.g there will always be a ‘)’ after a ‘(‘ or a newline after ‘:’.

Sources I recommend

- The Deep Learning Cookbook -> O'Riley, Douwe Osinga, 2018
- Fundamentals of Deep Learning -> O'Riley, Nikhil Buduma, 2017
- Natural Language Processing with Transformers -> O'Riley, Lewis Tunstall, 2022
- 3Blue1Brown Machine Learning Series -> Youtube
- Welch labs -> YouTube
- Attention is All you Need -> Google AI Research, 2017 (this paper introduces transformers)

Fin

What do you want to See next?

Any Qs?