EEP 596: Adv Intro ML || Lecture 10

Dr. Karthik Mohan

Univ. of Washington, Seattle

February 7, 2023

Last Time

- tSNE
- Agglomerative Clustering

Lectures and Programming Assignments

Week	Lecture Material	Assignment
1	Linear Regression	Housing Price Prediction
2	Classification	Spam classification (Kaggle)
3	Classification	Flower/Leaf classification
4	**Clustering**	MNIST digits clustering
5	Anomaly Detection	Crypto Prediction (Kaggle + P)
6	**Data Visualization**	Crypto Prediction (Kaggle + P)
	and Embeddings	
7	Deep Learning	Visualizing 1000 images
8	Deep Learning (DL)	ECG Arrythmia Detection
9	DL in NLP	TwitterSentiment Analysis (Kaggle + P)
10	DLs in Vision	TwitterSentiment Analysis (Kaggle + P)

Today

a. Agglomerative Clustering Wrap up
b. SVD and Dimensionality Reduction
c. Word2Vec and X2Vec

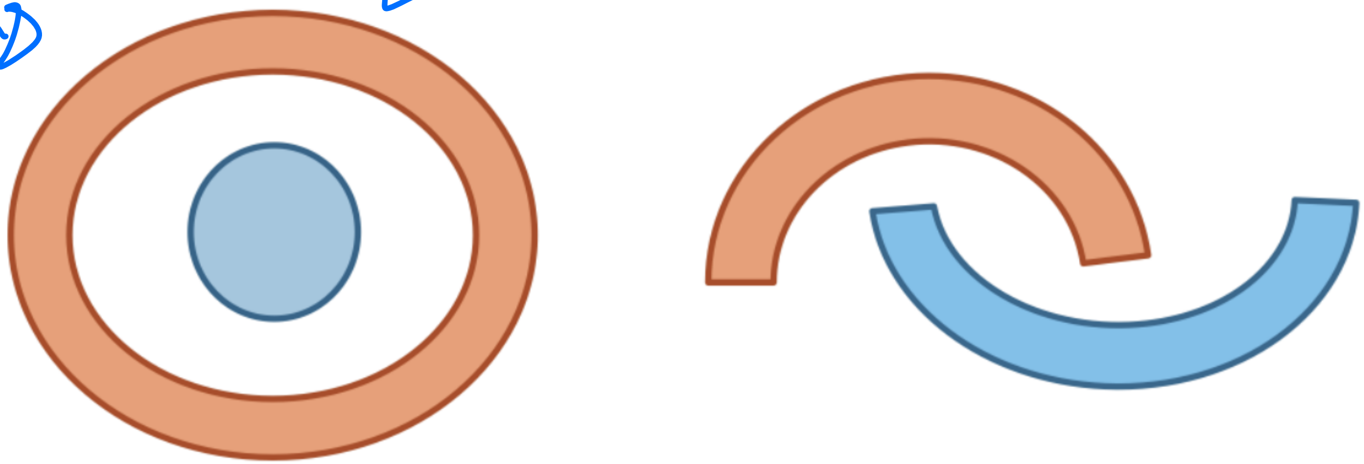
Agglomerative Clustering: Spiral and Donut!

With agglomerative clustering, we are now very able to learn weirder clusterings like

Agglomerative Clustering

KMeans

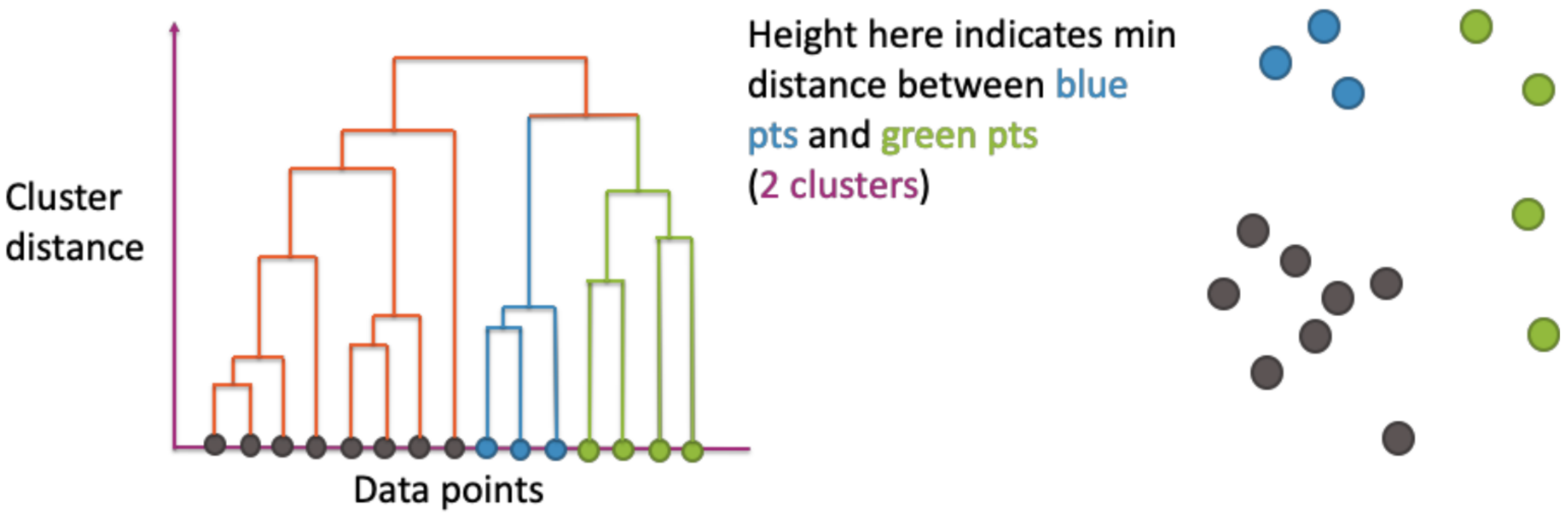
?



Dendrogram

x-axis shows the datapoints (arranged in a very particular order)

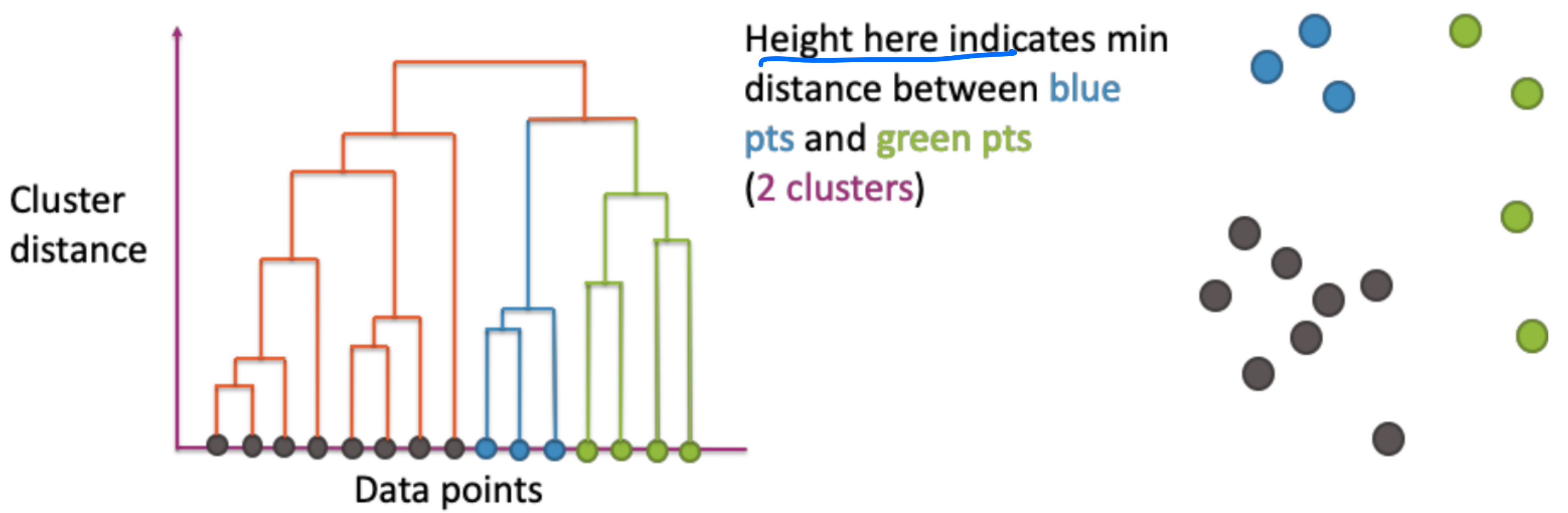
y-axis shows distance between pairs of clusters



Dendrogram

x-axis shows the datapoints (arranged in a very particular order)

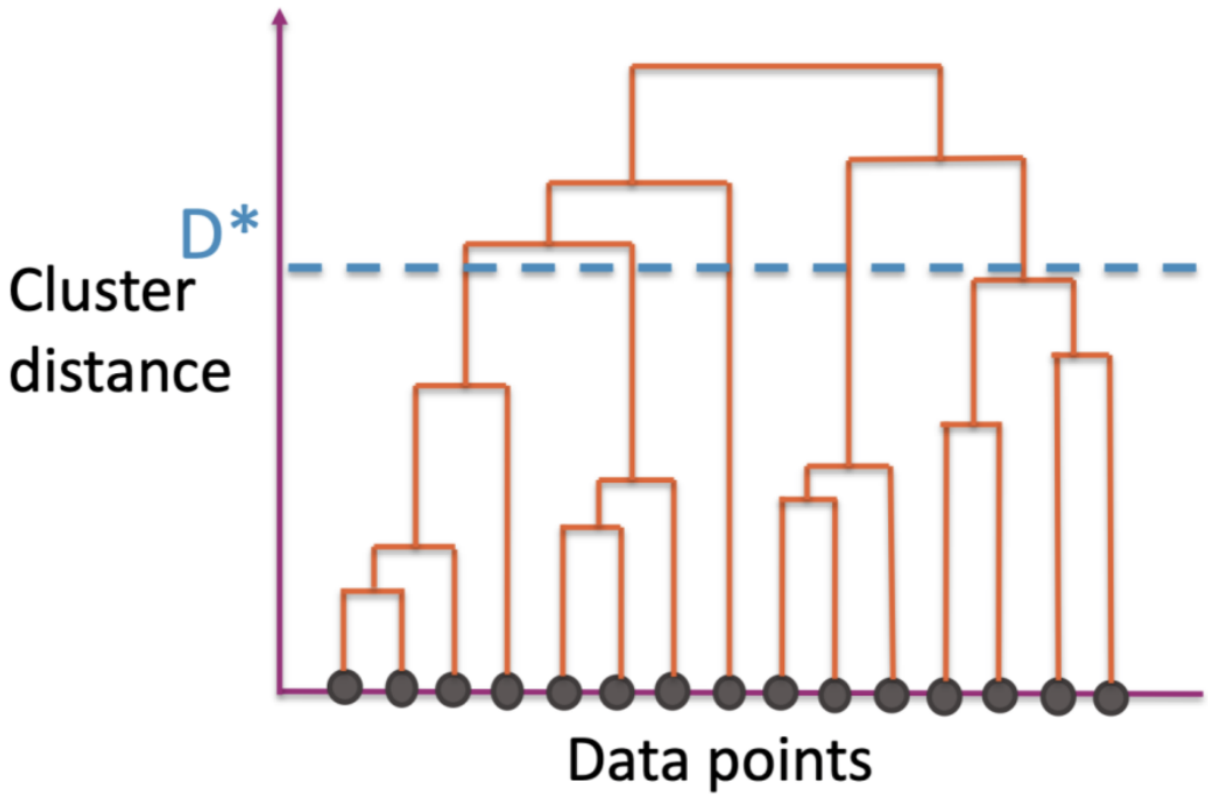
y-axis shows distance between pairs of clusters



Cut Dendrogram

Choose a distance D^* to "cut" the dendrogram

- Use the largest clusters with distance $< D^*$
- Usually ignore the idea of the nested clusters after cutting

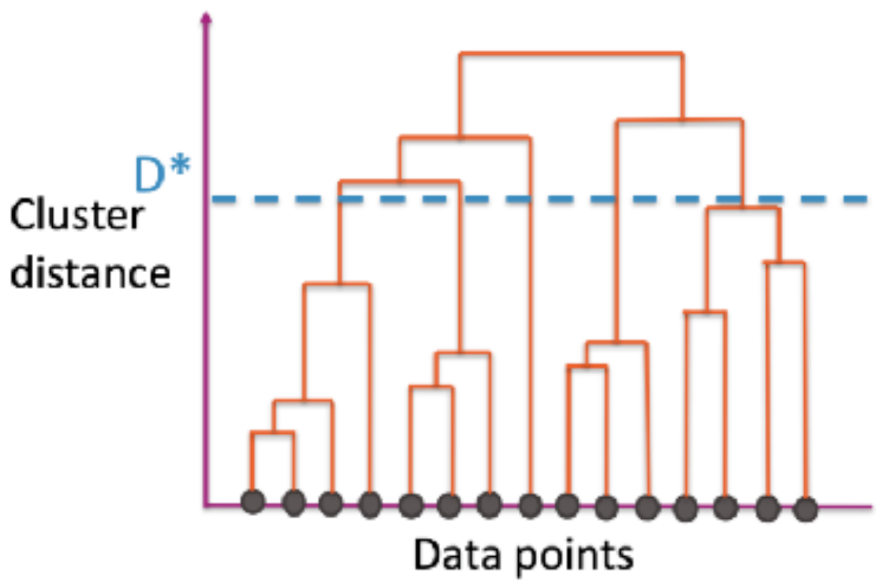


Dendrogram ICE

ICE #2

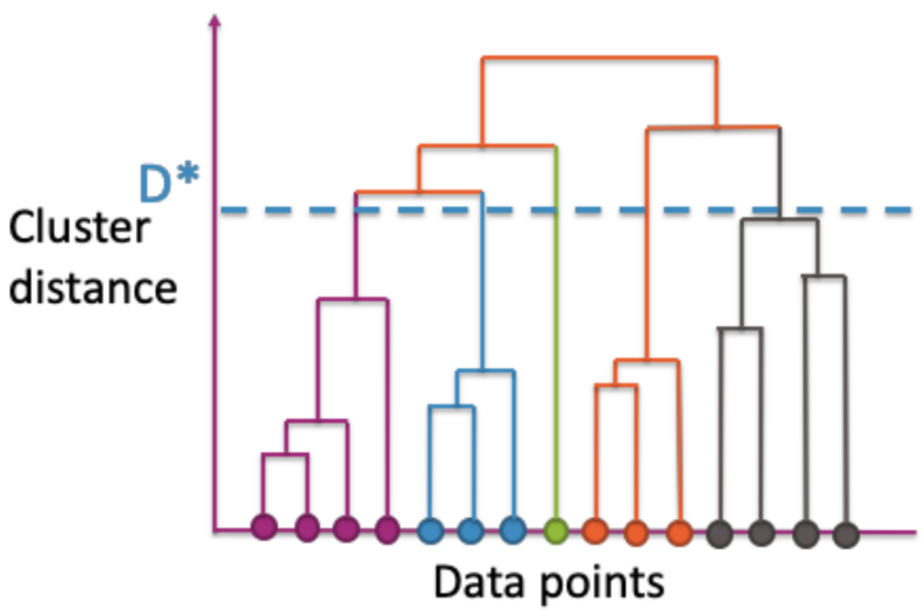
How many clusters would we have if we use this threshold to cut?

a. 4
b. 5
c. 6
d. 7



Cut Dendrogram

Every branch that crosses D^* becomes its own cluster



Agglomerative Clustering — Hyper-parameters

For agglomerative clustering, you need to make the following choices:

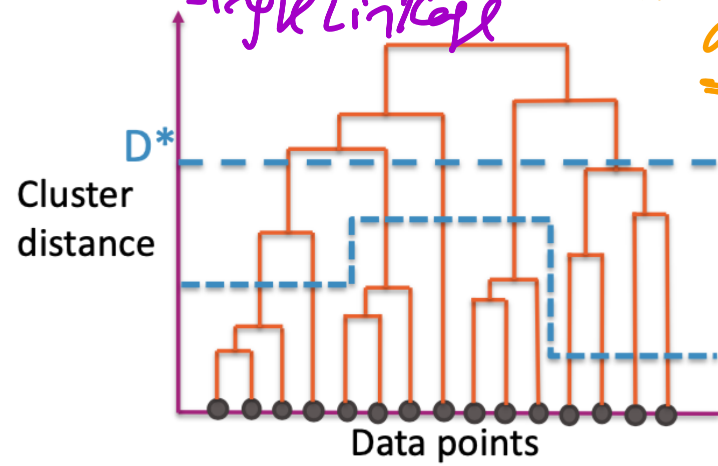
- Distance metric $d(x_i, x_j)$ → Between two data pts x_i & x_j
- Linkage function → Distance between 2 clusters
  1) - Single Linkage:
    $$\min_{x_i \in C_1, x_j \in C_2} d(x_i, x_j)$$
  2) - Complete Linkage:
    $$\max_{x_i \in C_1, x_j \in C_2} d(x_i, x_j)$$
  3) - Centroid Linkage
    $$d(\mu_1, \mu_2)$$
  - Others
- Where and how to cut dendrogram

$d(x_i, x_j)$

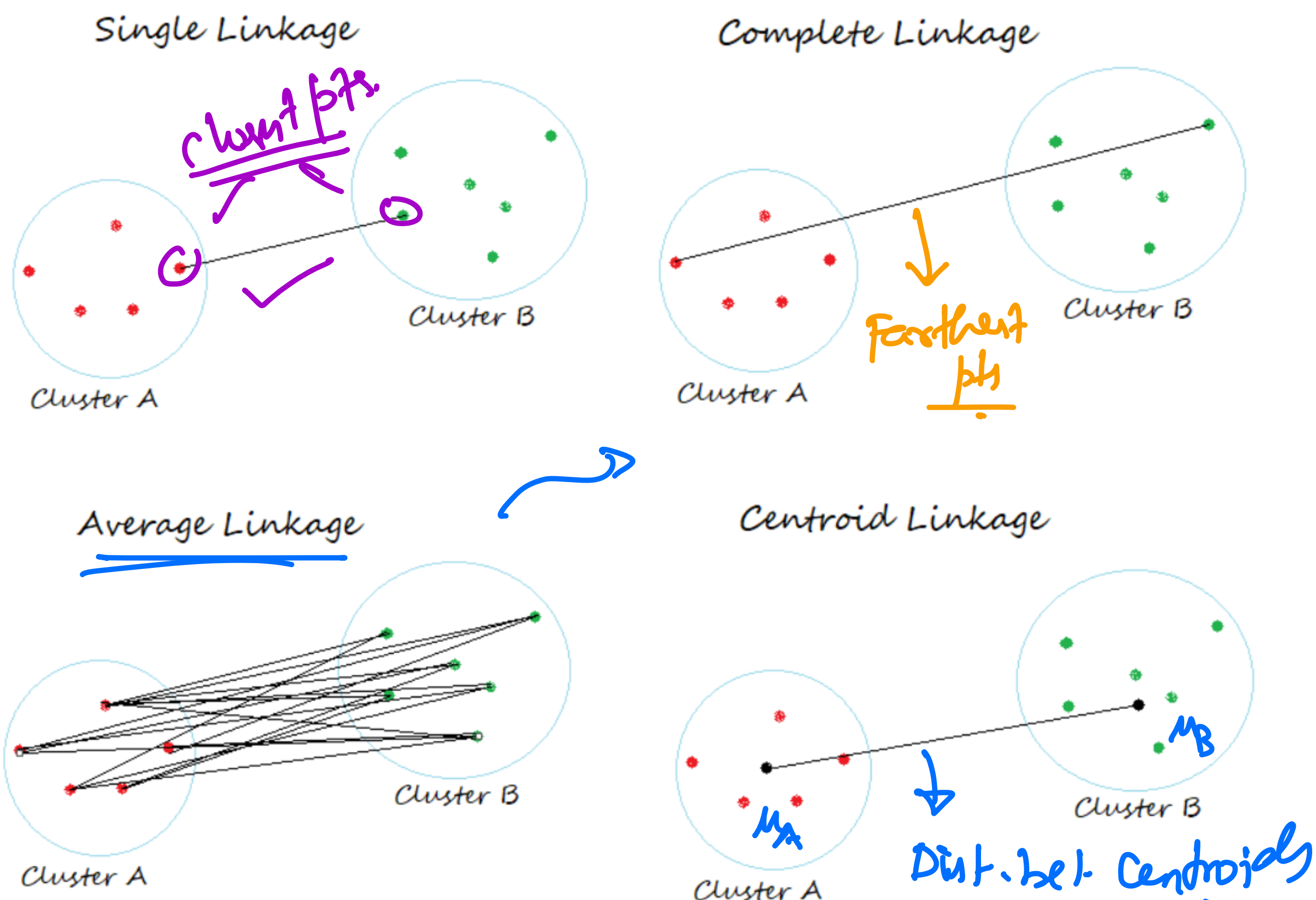
μ_1 x_i x_j μ_2

Single Linkage

Centroid distance



Linkage examples



Centroid Linkage / Avg. Linkage < Complete Linkage

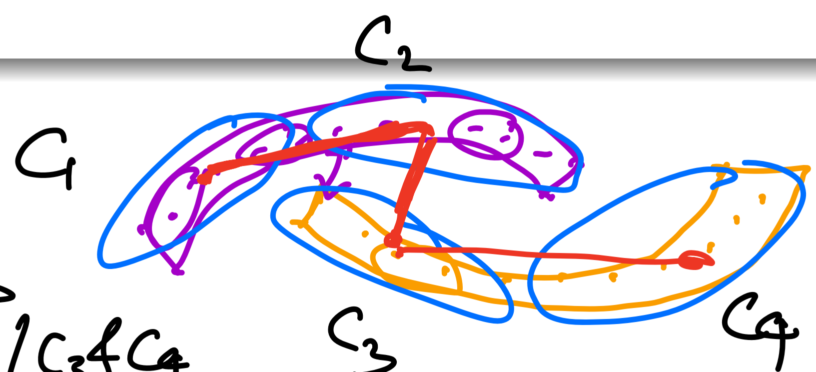
> Single Linkage

Dendrogram ICE

ICE #1

Which linkage function is more likely to detect spiral clusters?

a. Single Linkage
b. Centroid Linkage
c. Complete Linkage
d. Any Linkage



Likely merge

Single Linkage:- C_2 & C_3
Centroid —||— :- C_1 & C_2 / C_3 & C_4
Complete —||— :- —||—

Centroid Linkage Applied to Spiral

With agglomerative clustering, we are now very able to learn weirder clusterings like

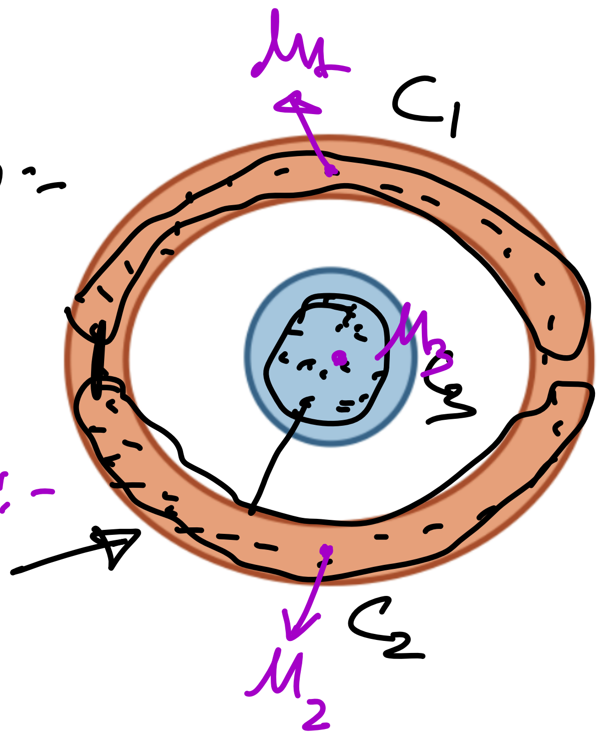
Likely Merge

Single Linkage :-

C_1 & C_2

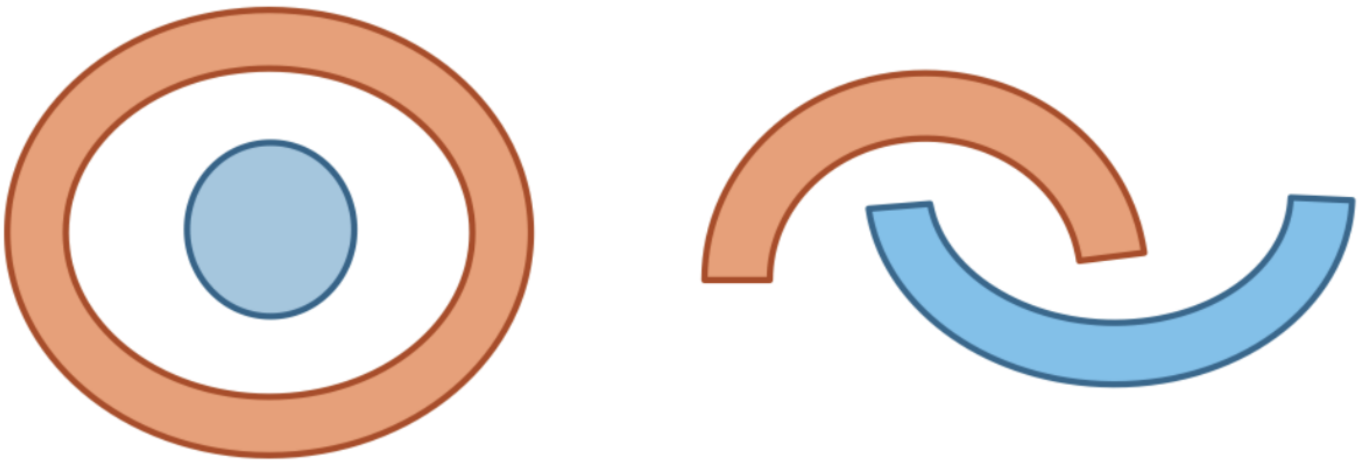
Centroid Linkage :-

C_1 & C_3 / C_3 & C_2

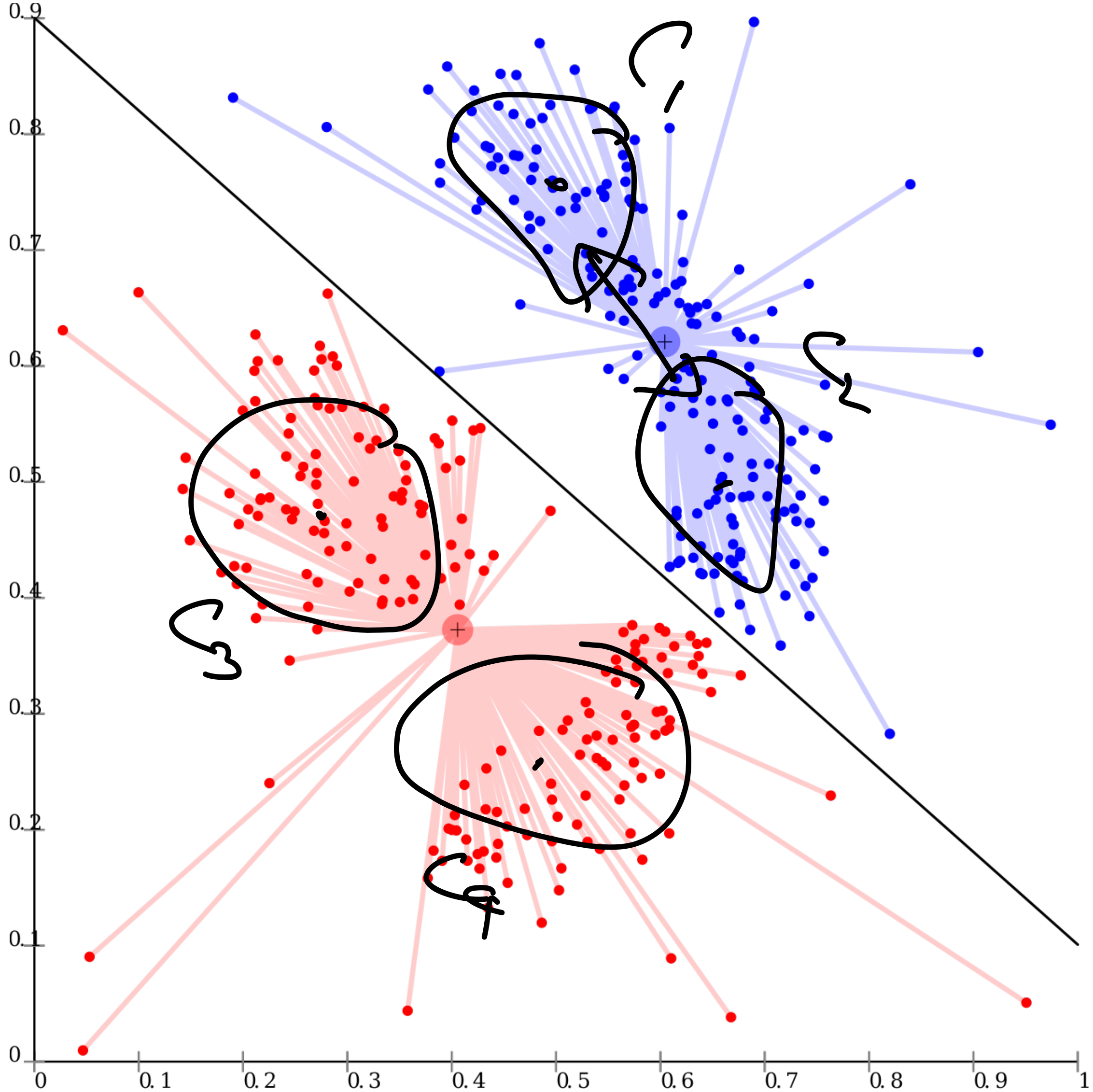


Single Linkage Applied to Spiral

With agglomerative clustering, we are now very able to learn weirder clusterings like



Where Centroid Linkage Works!



Dendrogram

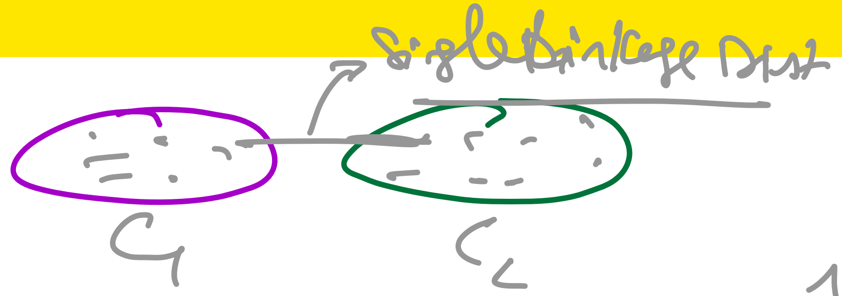
For visualization, generally a smaller # of clusters is better

For tasks like outlier detection, cut based on:

- Distance threshold
- Or some other metric that tries to measure how big the distance increased after a merge

No matter what metric or what threshold you use, no method is "incorrect". Some are just more useful than others.

Dendrogram



Computation

1. All pair dist.
$\binom{n}{2}$ pairs $\rightarrow O(n^2)$

2. Find the smallest dist. $\Rightarrow$ Sorting
$O(n^2 \log(n^2))$
$= O(n^2 \log(n))$

Computing all pairs of distances is pretty expensive!

- A simple implementation takes $\mathcal{O}(n^2 \log(n))$

Can be much implemented more cleverly by taking advantage of the **triangle inequality**

- "Any side of a triangle must be less than the sum of its sides"

Best known algorithm is $\mathcal{O}(n^2)$

Comparison of Clustering Algorithms

$N \approx 10k$

$d \approx 500$ $k \approx 10$

Quick comparison

	k-means	Agglomerative Clustering
Computation	$O(Ndk)$	$O(N^2d)$
Type	Spherical	Arbitrary shapes

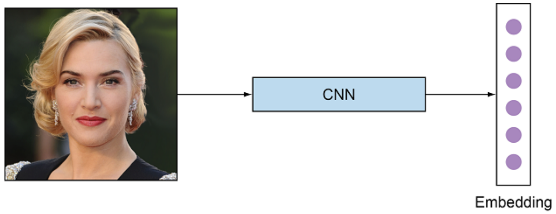
Few more points..

a. Weigh computational complexity with complexity of clustering - kmeans vs agglomerative
b. Agglomerative distance choices yield different sets of clusters (single linkage vs centroid)
c. Clustering in practice is an art
d. However, quality of clustering can be evaluated - E.g. through Dunn Index!

Dimensionality Reduction and Embeddings

Basic Idea of Embeddings

Can you capture information in an image concisely, in a vector? Perhaps a 500x500 pixel image can be reduced to a 256 dimensional vector, z and z may have most of the information about the original image that can help you make predictions on the image - Say image classification or object detection or image captioning! Say you have 300k words in a vocabulary. Can you capture the semantics and information contained in each word concisely, maybe through a 512 dimensional vector? Perhaps this representation can help you solve a word riddle such as: *"What is King - Man + Woman?"* Concise representations of this kind are referred to as Embeddings.



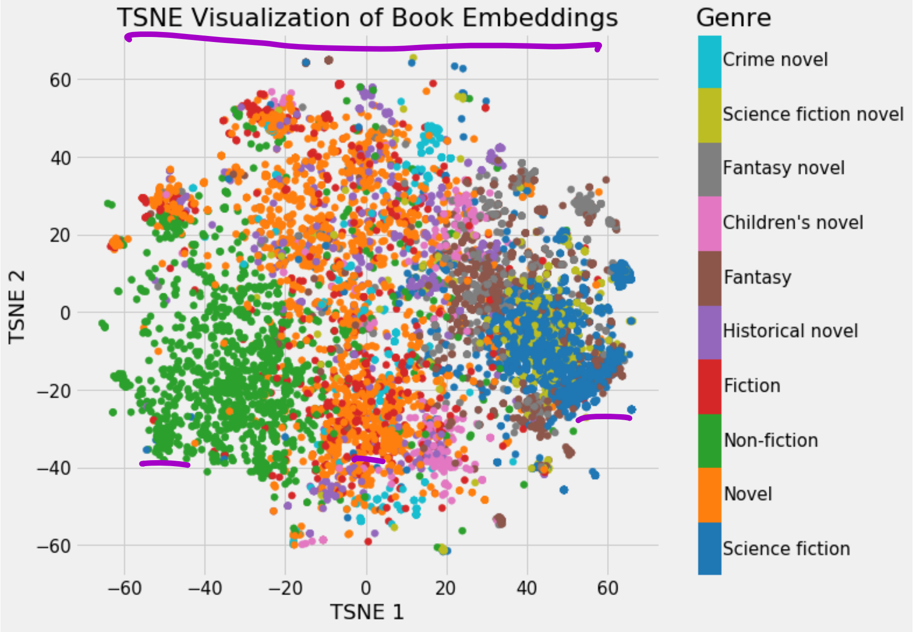
$z \in \mathbb{R}^{256}$

Dimensionality Reduction and Embeddings

Today

Today we will learn about embeddings learned through SVD and one specifically for words as well. Embeddings also overlap with the idea of dimensionality reduction - Which can also serve the purpose of compressing data. **What's an application of data compression that we use on a daily basis?**

TSNE :-
specifically for
2-D Data visualization
("2-D Embedding")

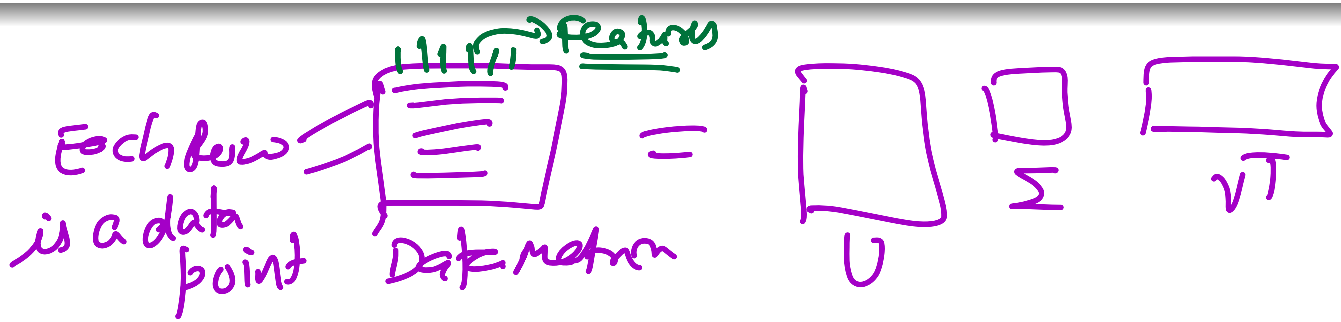


SVD - Classic Method

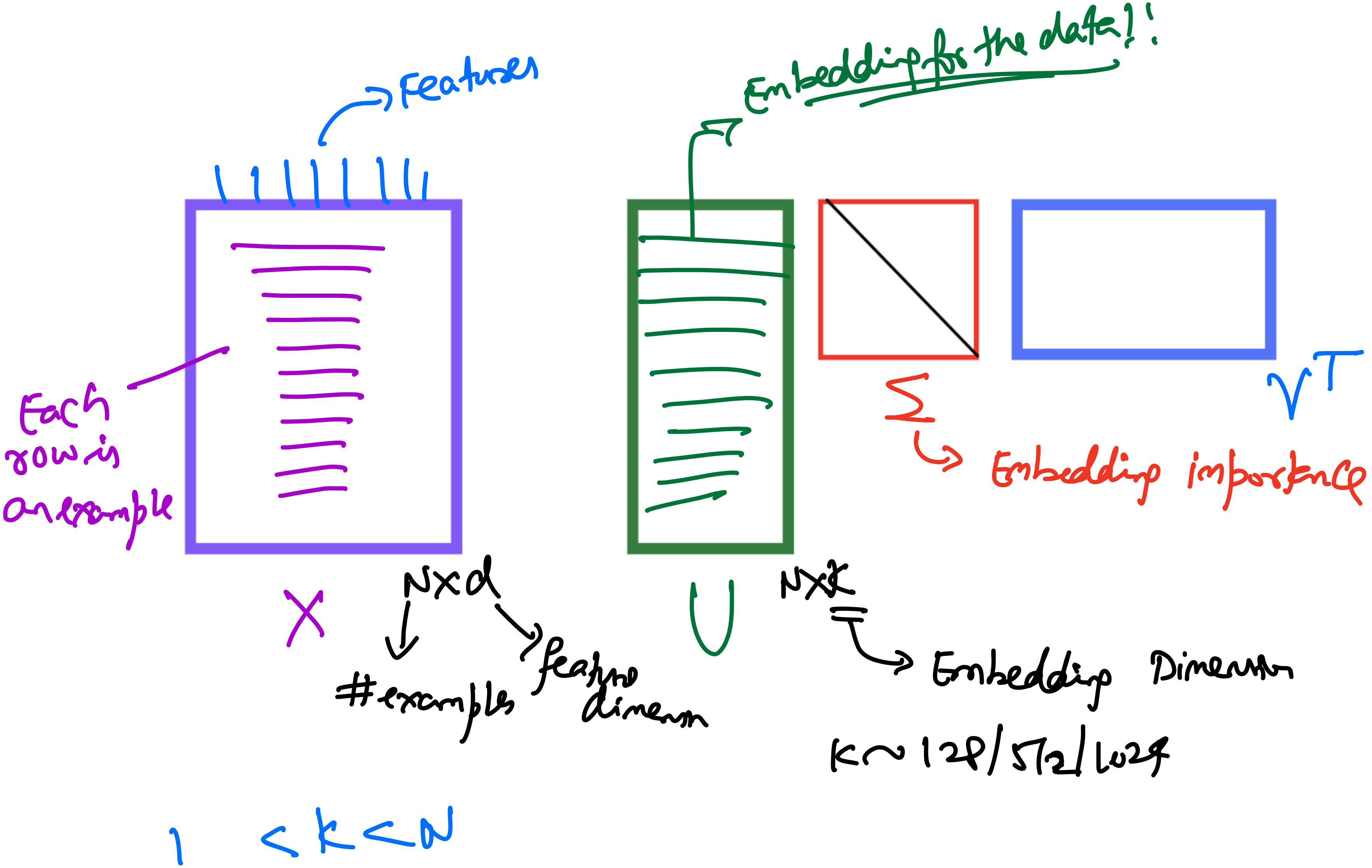
SVD

Every matrix $X \in \mathcal{R}^{m \times n}$ has a Singular Value Decomposition:

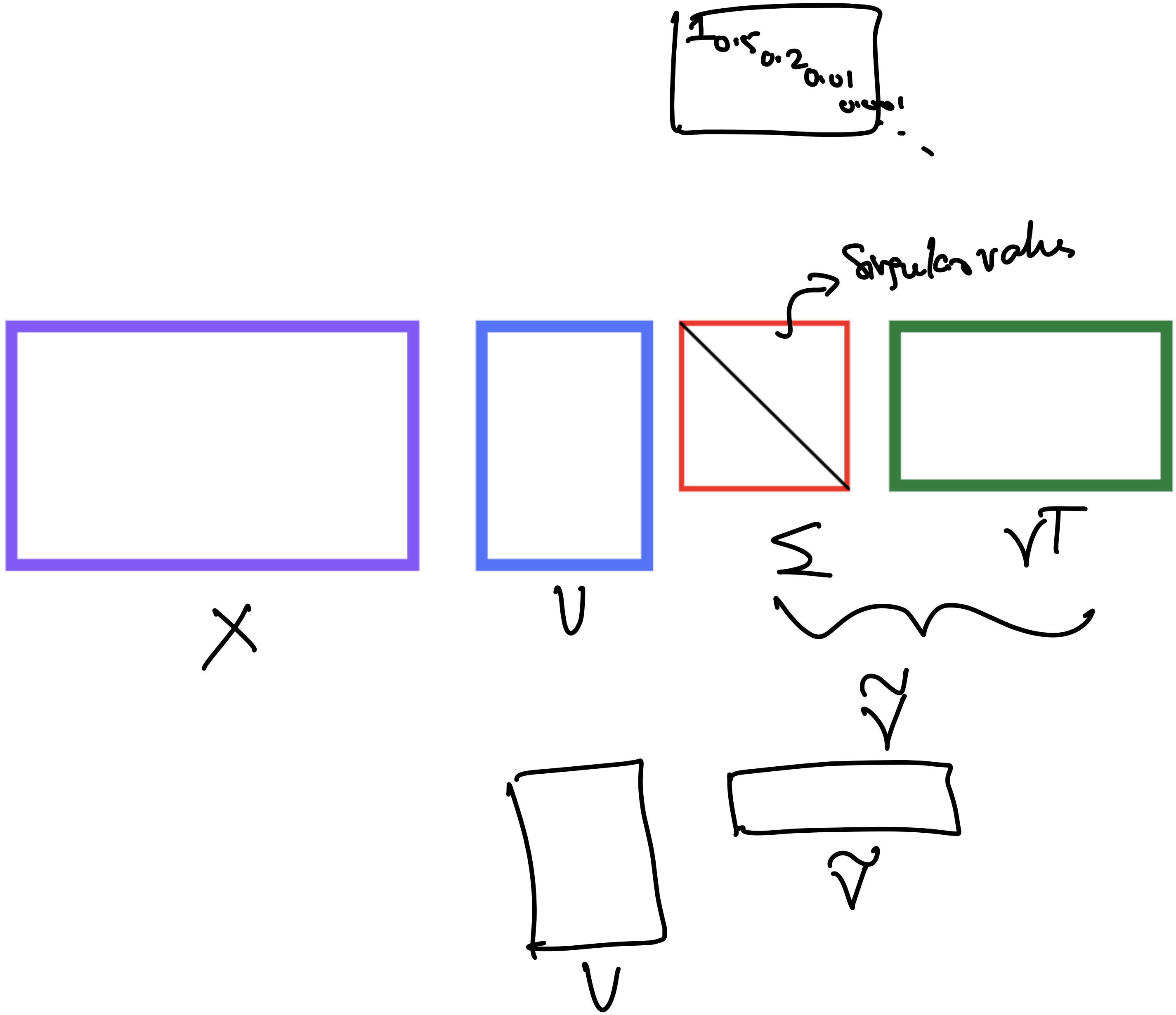
$$X = U \Sigma V^T$$



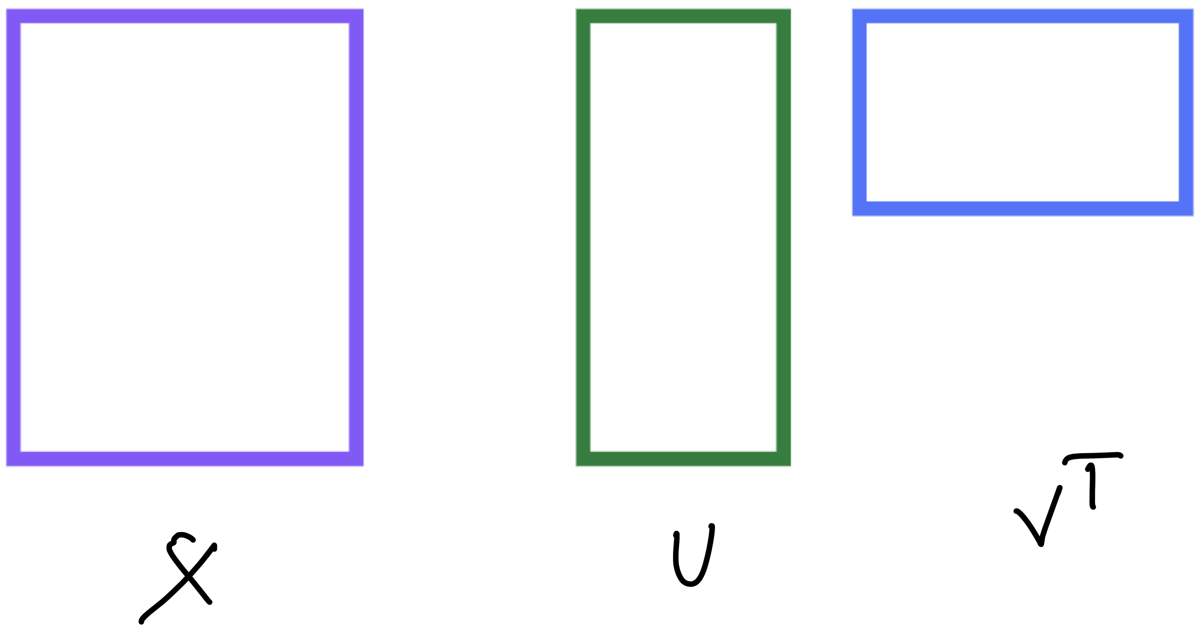
SVD Example


Features
Embedding for the data!!
Each row is an example
X
Nxd
#examples
feature dimension
U
Nxk
Embedding Dimension
k~128/512/1024
Σ
Embedding importance
V^T
1 < k < N

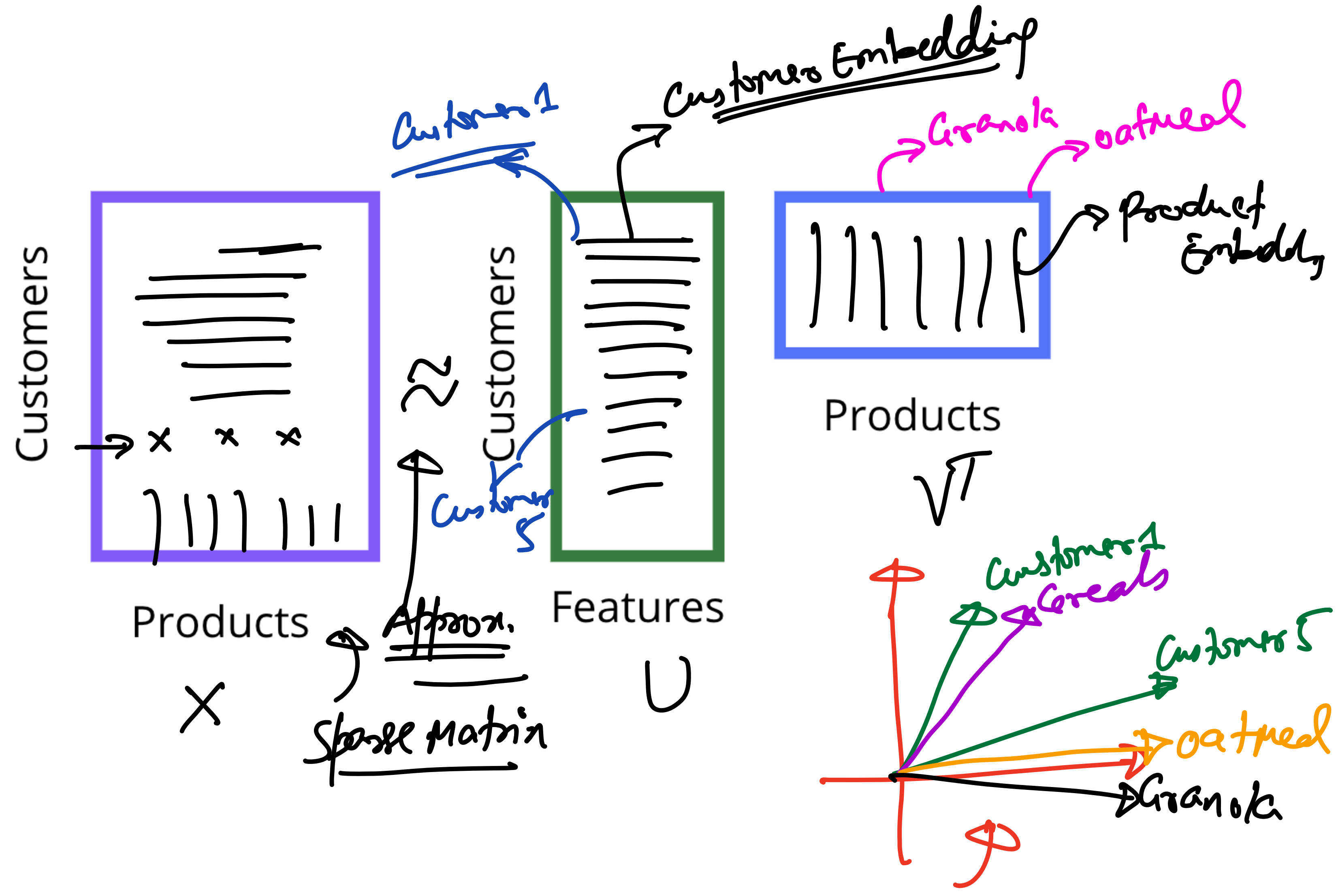
SVD Example



SVD Example

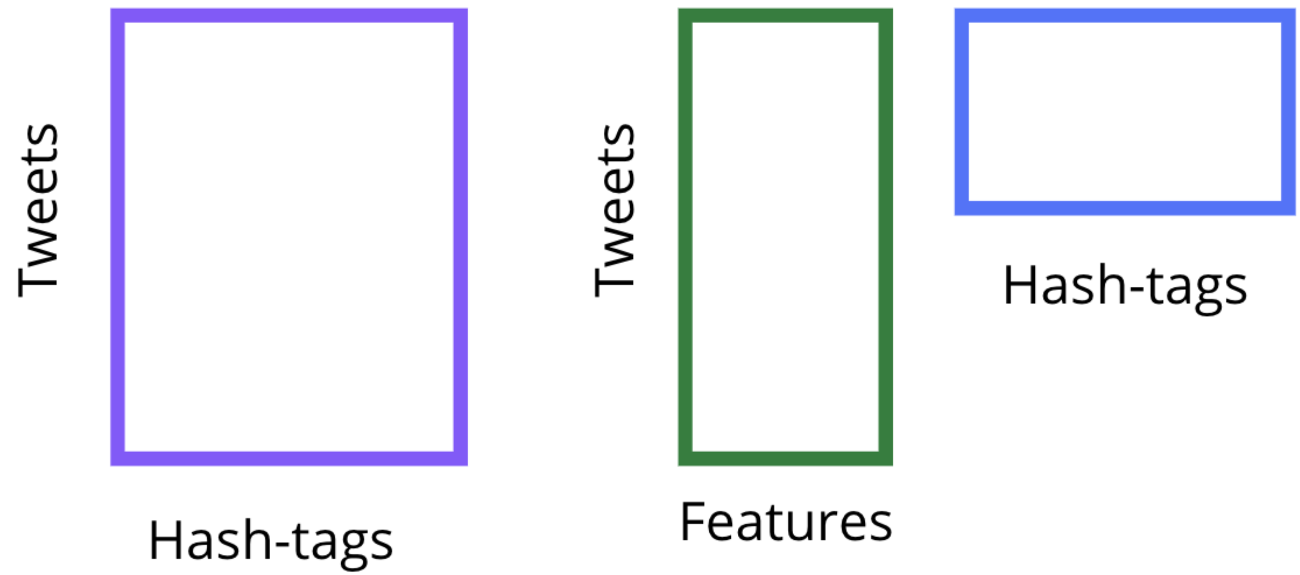


SVD Example



Customers
Products
X
≈
Approx.
Sparse Matrix
Customer 1
Customer Embedding
Customer 5
Customers
Features
U
Granola
oatmeal
Product Embedding
Products
V^T
Customer 1
Cereals
Customer 5
oatmeal
Granola

SVD Example



Tweet: - . . . - -

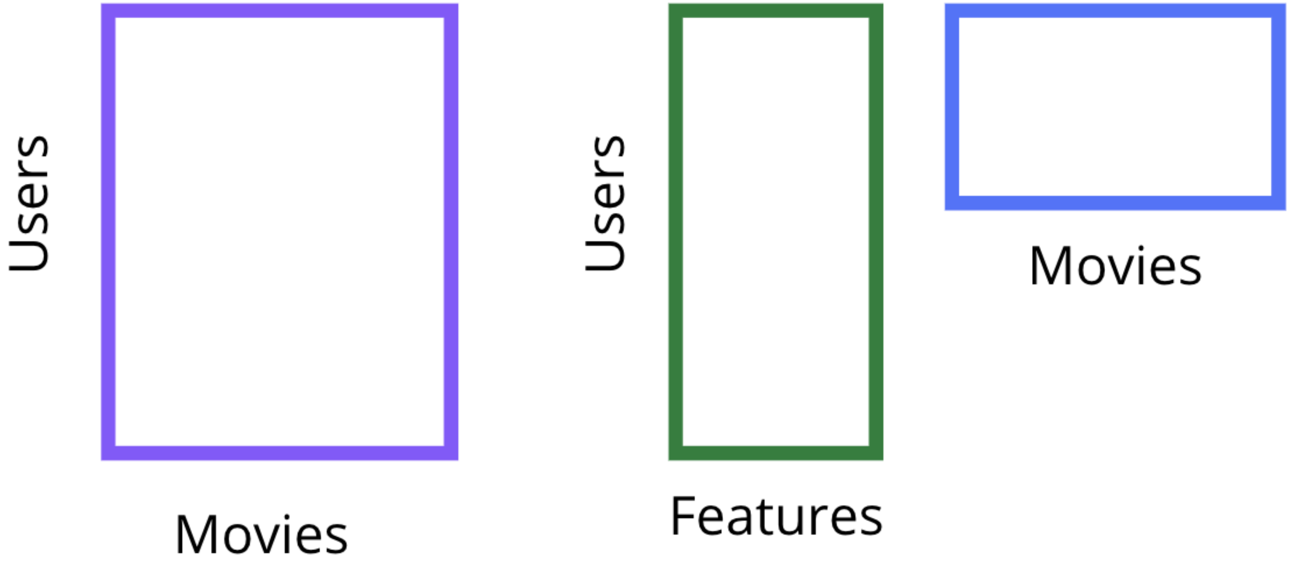
#ml

first hashtag

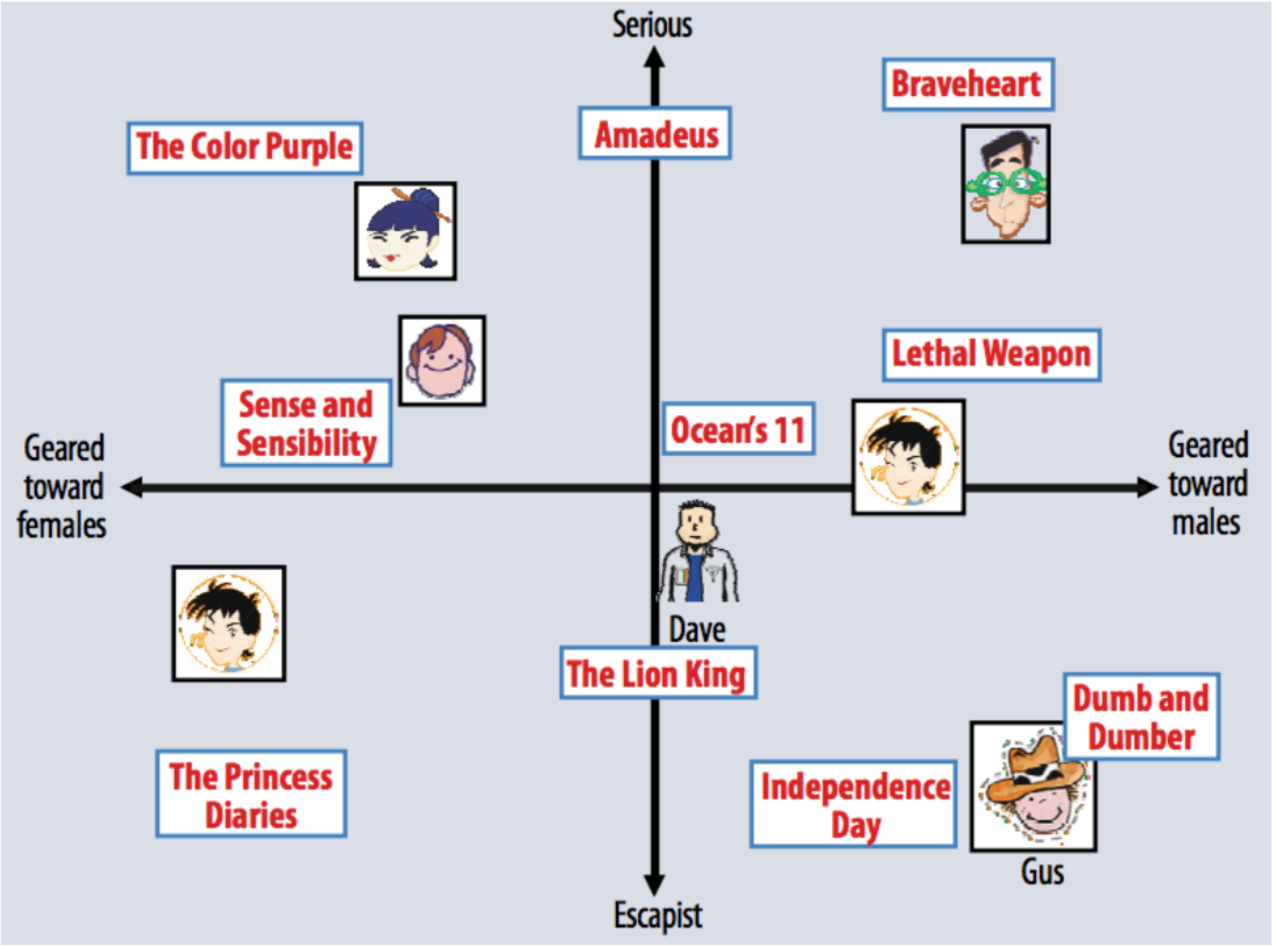
#ai #machine learning . . .

Twitter suggests

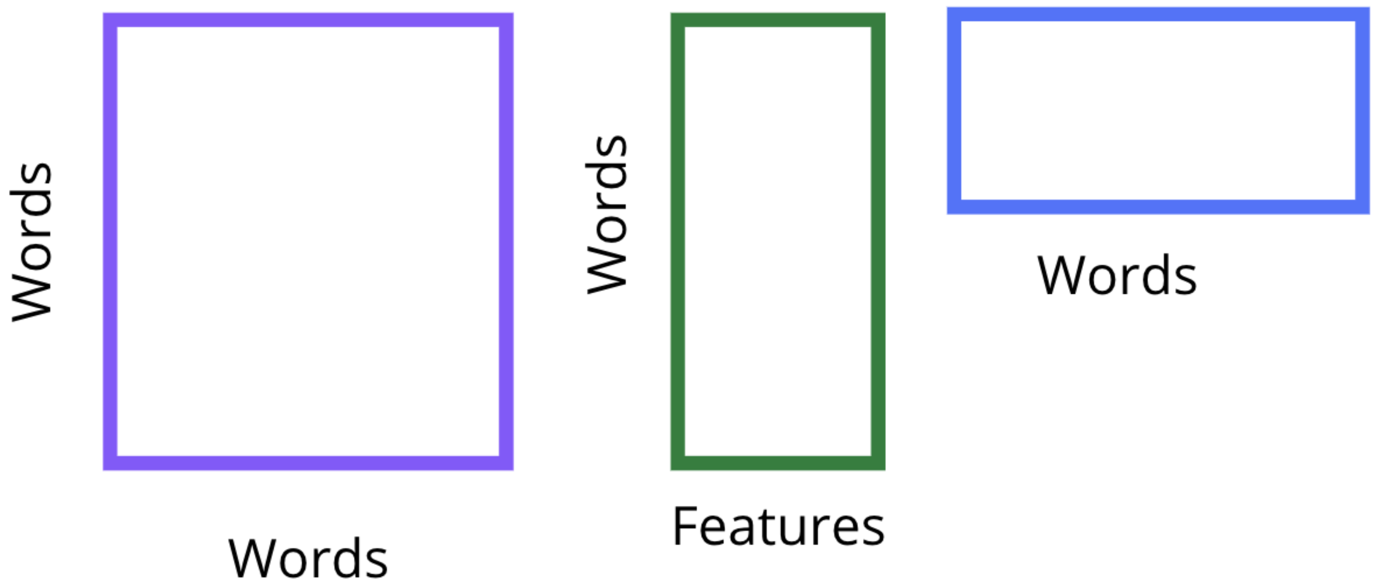
SVD Example



SVD Example


Serious
Braveheart
Amadeus
The Color Purple
Lethal Weapon
Sense and Sensibility
Ocean's 11
Geared toward females
Geared toward males
Dave
The Lion King
Dumb and Dumber
The Princess Diaries
Independence Day
Gus
Escapist

SVD Example



Word co-occurence

Example Sentence

I like NLP. I enjoy flying. I like deep learning.

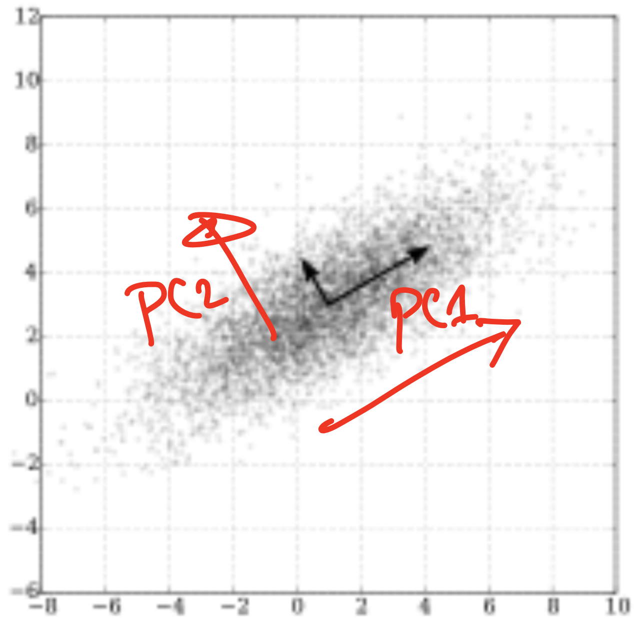
$$X = \begin{array}{c|cccccccc} & I & like & enjoy & deep & learning & NLP & flying & . \\ \hline I & 0 & 2 & 1 & 0 & 0 & 0 & 0 & 0 \\ like & 2 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ enjoy & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ deep & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ learning & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ NLP & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ flying & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ . & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{array}$$

co-occurence matrix

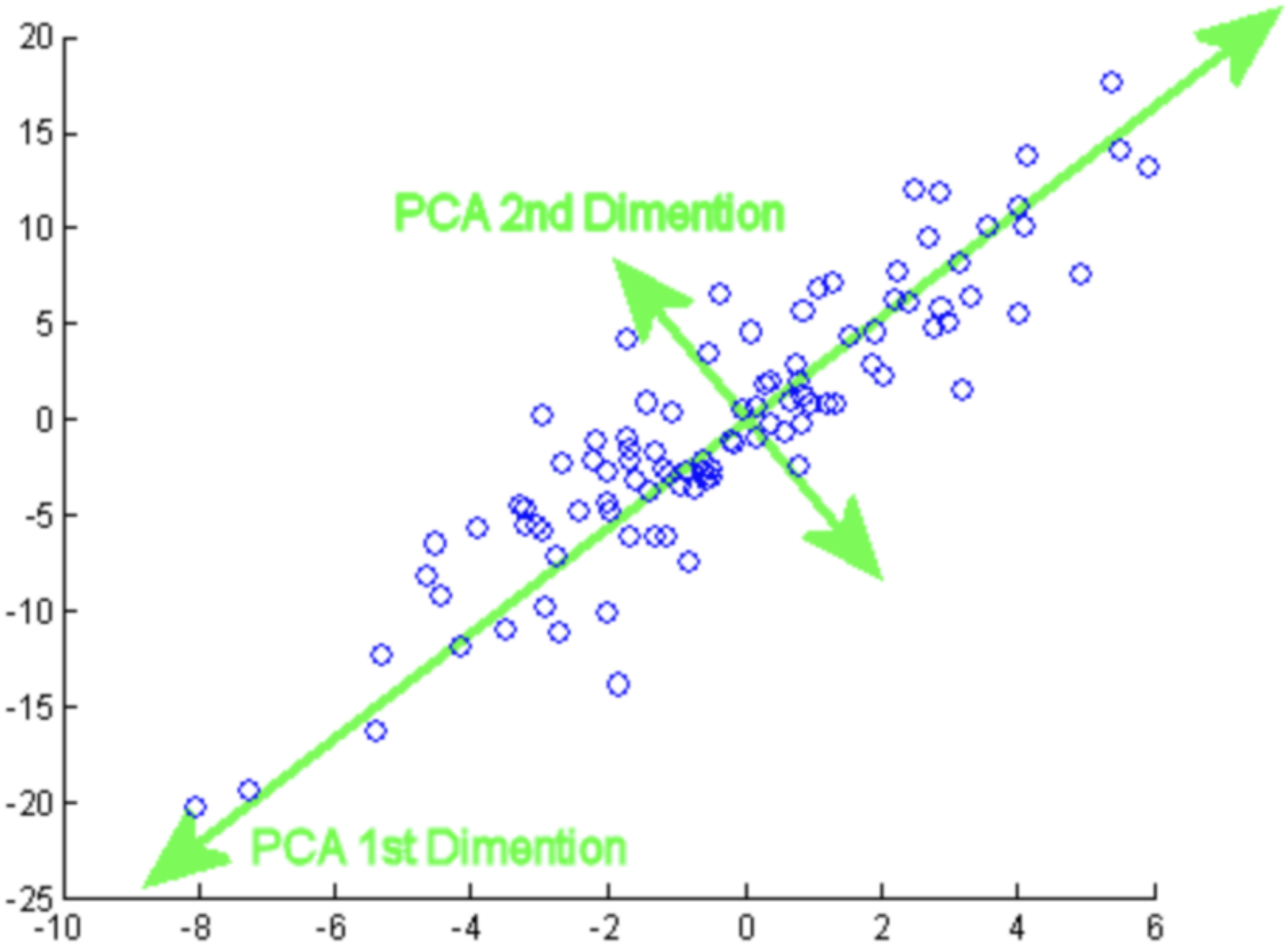
Other Popular Low-dimensioanl Embeddings

PCA

Principal components Analysis (based on SVD) tells us the direction of maximum variance in the data!



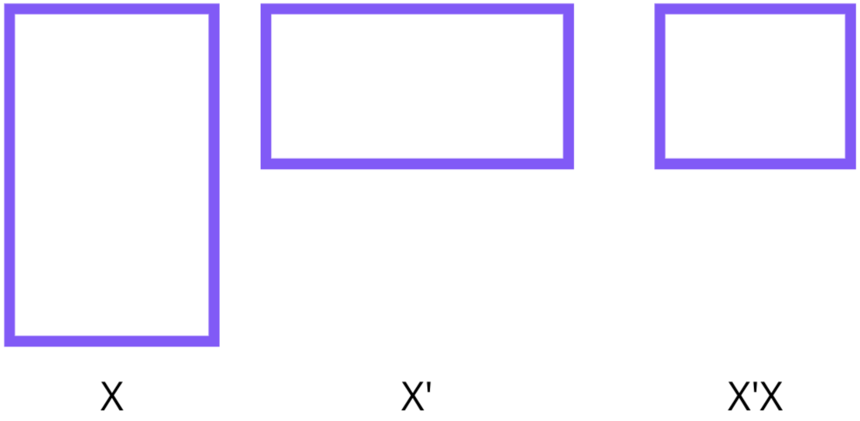
PCA


PCA 2nd Dimention
PCA 1st Dimention
20
15
10
5
0
-5
-10
-15
-20
-25
-10
-8
-6
-4
-2
0
2
4
6

PCA

PCA model

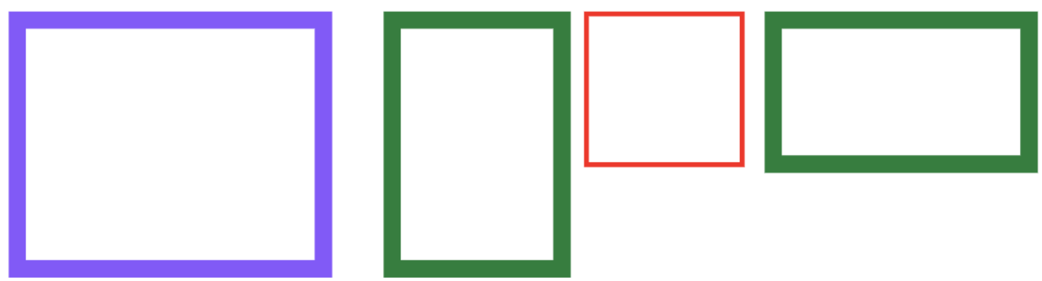
Obtained through the eigen-decomposition of the sample covariance matrix



SVD and PCA

PCA model

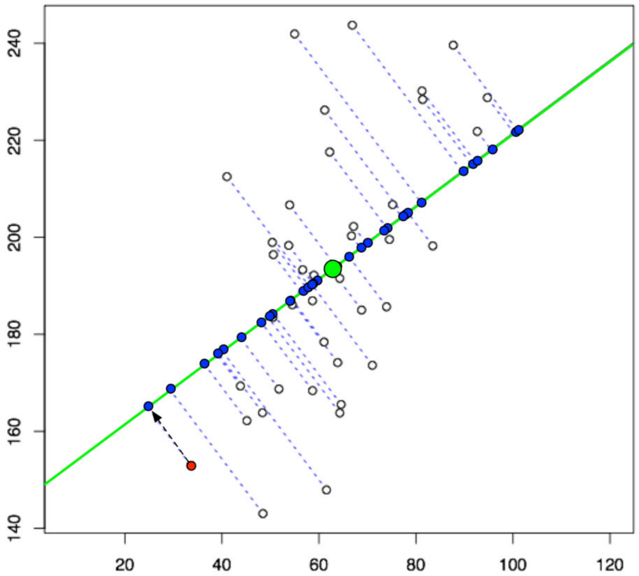
SVD on the covariance matrix is the same as Eigen-decomposition of covariance matrix.



PCA Embeddings

PCA Embeddings

Projecting data onto the PCA directions also gives us low-dimensional embeddings.



Principal Components and Embeddings!

$X = U\Sigma V^T$

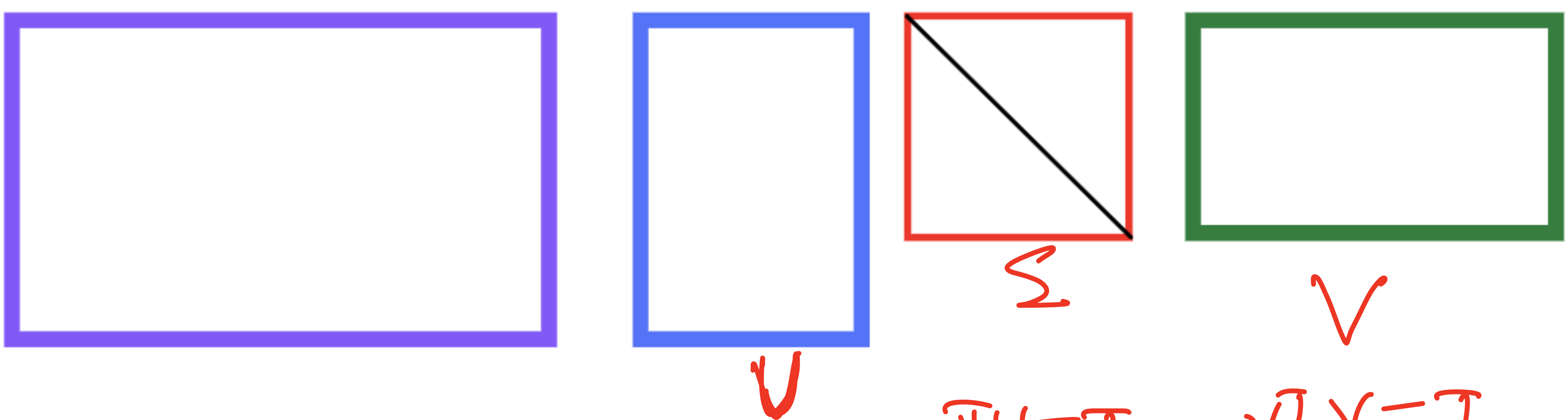
$X^T = V\Sigma U^T$

$X^T X = V\Sigma \underbrace{U^T U}_{I} \Sigma V^T$

$= V\Sigma^2 V^T$

Principal Components

If $X^T X = V\Sigma^2 V^T$ is the co-variance matrix, then V represents the principal components and $V^T X^T$ represents the embeddings or compressed representation of the data points!



$U^T U = I$ $\quad$ $V^T V = I$

ICE #3

$X_{N\times d} \approx U_{N\times k} \Sigma_{k\times k} V^T_{k\times d}$

SVD manipulations

Let $X = U\Sigma V^T$. The projection of the data onto the principal components is given by $V^T X^T$. If X is $N \times d$, what's the dimension of the projection matrix with 3 principal components?

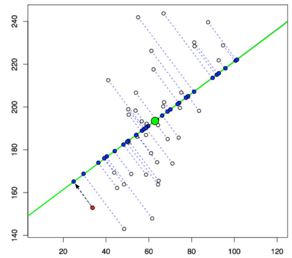
(a) $3 \times d$
(b) $3 \times N$
(c) $N \times 3$
(d) $d \times N$

$k = 3$!

$V^T X^T = \underbrace{V^T V}_{I} \Sigma U^T$

$= \Sigma U^T$

ICE #4



Projection Dimension

Consider the figure above. Let $V_1 \in \mathcal{R}^d$ be the first principal component and the project of a data point $x \in \mathcal{R}^d$ onto V_1 is given by $V_1^T x$. Now let $V \in \mathcal{R}^{d \times k}$ be the entire set of principal components. When $x \in \mathcal{R}^d$ gets projected onto all the principal components, what's the dimension of the projected vector?

1. d
2. k
3. N
4. 1

ICE #5

Images PCA (Work in groups of 2)

If you have 1000 face images and did a PCA on these images and found that 10 Eigen faces would be sufficient to reconstruct the images accurately. You stored compressed representations of the images on your laptop and to reconstruct the image, you send it to a server that then gives you back a re-constructed image. What would be the compression factor for the compressed representation you have on your laptop and obtained from PCA? Assume that each image is 1000×1000 pixels?

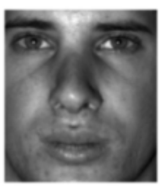
a. 1000x
b. 10000x
c. 100000x
d. 1MMx

PCA for Images

Eigen-face

Eigen-decompont

$X^T X = V \Sigma^2 V^T$



PCA for Images - Eigen Faces

original face Reconstructed face

PC1 PC2 PC3 PC4 PC5 PC6

1) 2) 3) 4)

x_4 $\hat{x}_4$ v_1 v_2 v_3 v_4 v_5 v_6

Embedding :- α α_1 α_2 α_3 α_4 α_5 α_6

$x_4 \to$ Image

$$\hat{x}_4 = \alpha_1 v_1 + \alpha_2 v_2 + \alpha_3 v_3 + \vdots + \alpha_6 v_6$$

PCA for Images - Re-construction

original

5 PC

20 PC

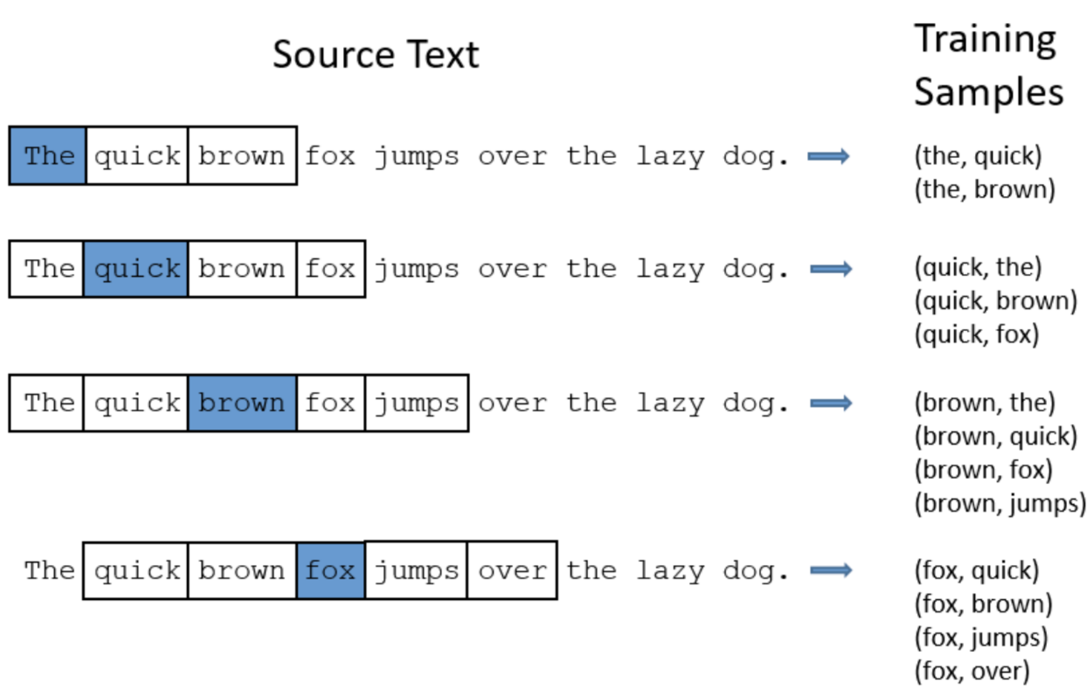
SVD / PCA - One main Issue

1. Not scalable!
2. Imagine millions of movies and millions of people watching it - How do you compute the SVD for it?
3. Complexity is $O(N^2k)$ where k is the latent dimension of the matrix.
   $\to k = 10$ or 20
4. Not very flexible
5. Also not robust to outliers
6. Uses a bi-linear model instead of a non-linear model

Word2Vec

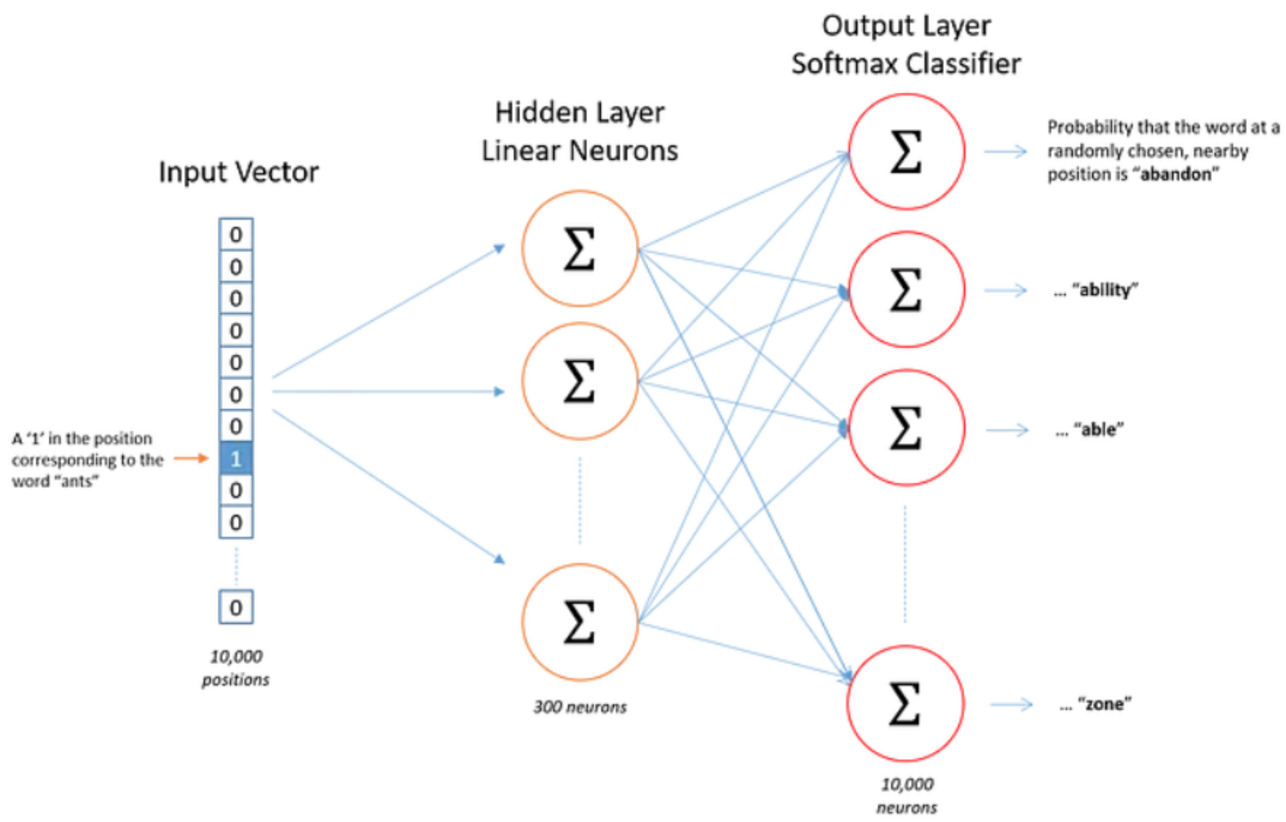
Skip Gram Model

Is based on the skip-gram model! How is training done? It's semi-supervised!!

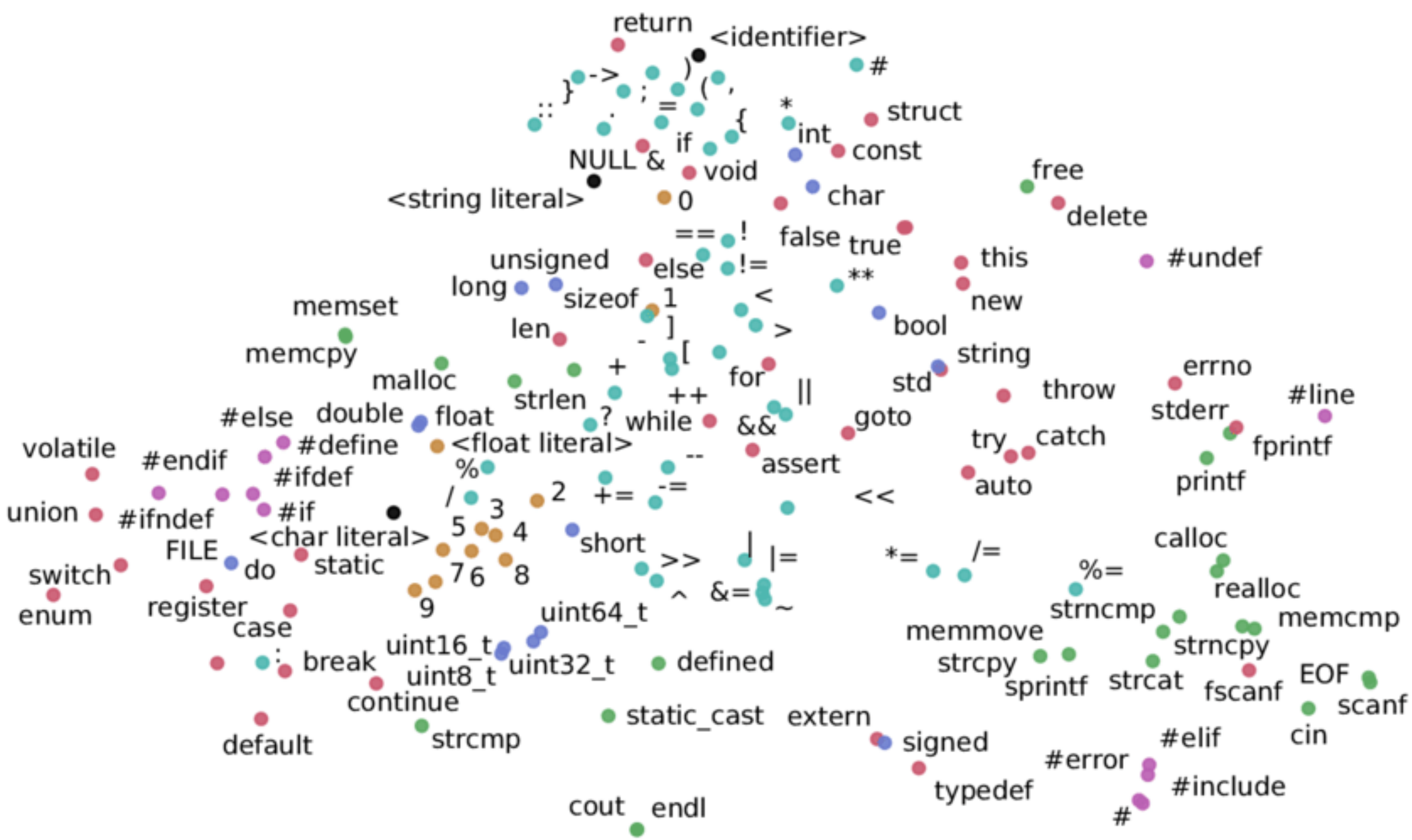


Word2Vec

Architecture

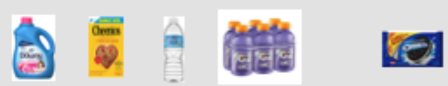


Word2Vec representation



Product2Vec

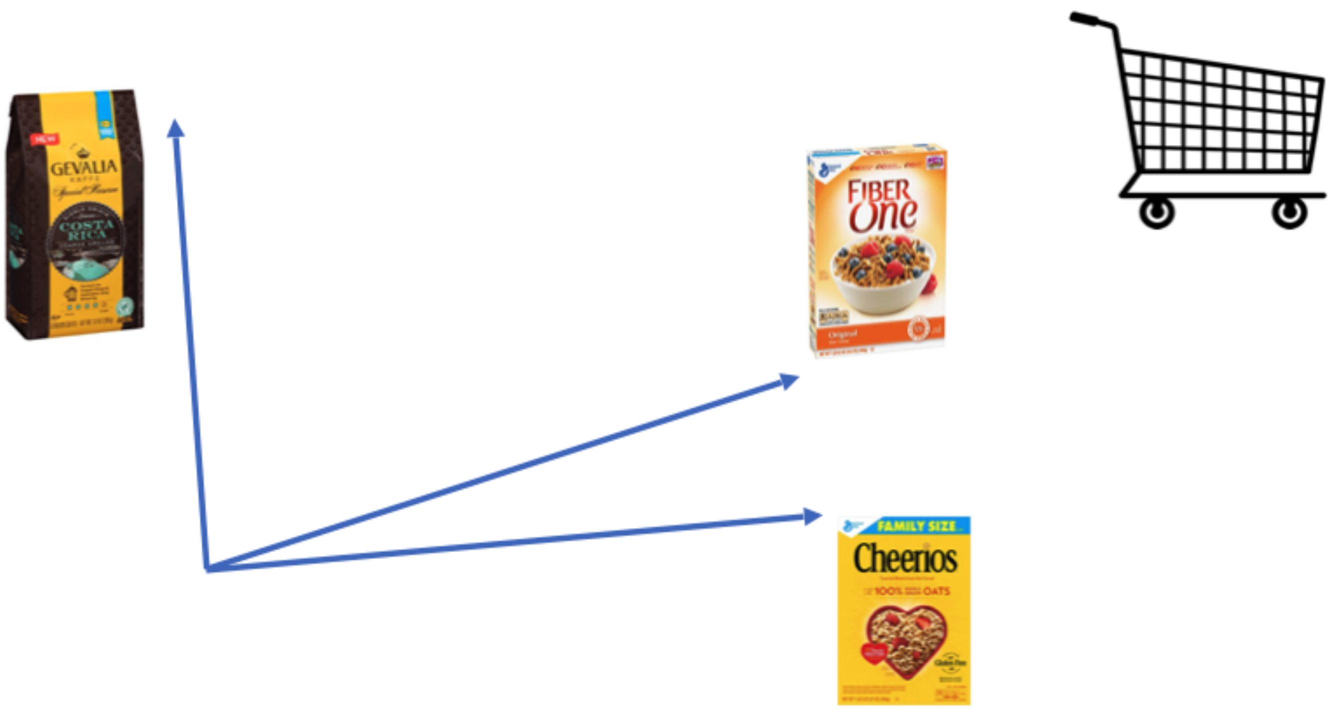
Represent products in product space with a large matrix of embedding coordinate vectors ***"L"***



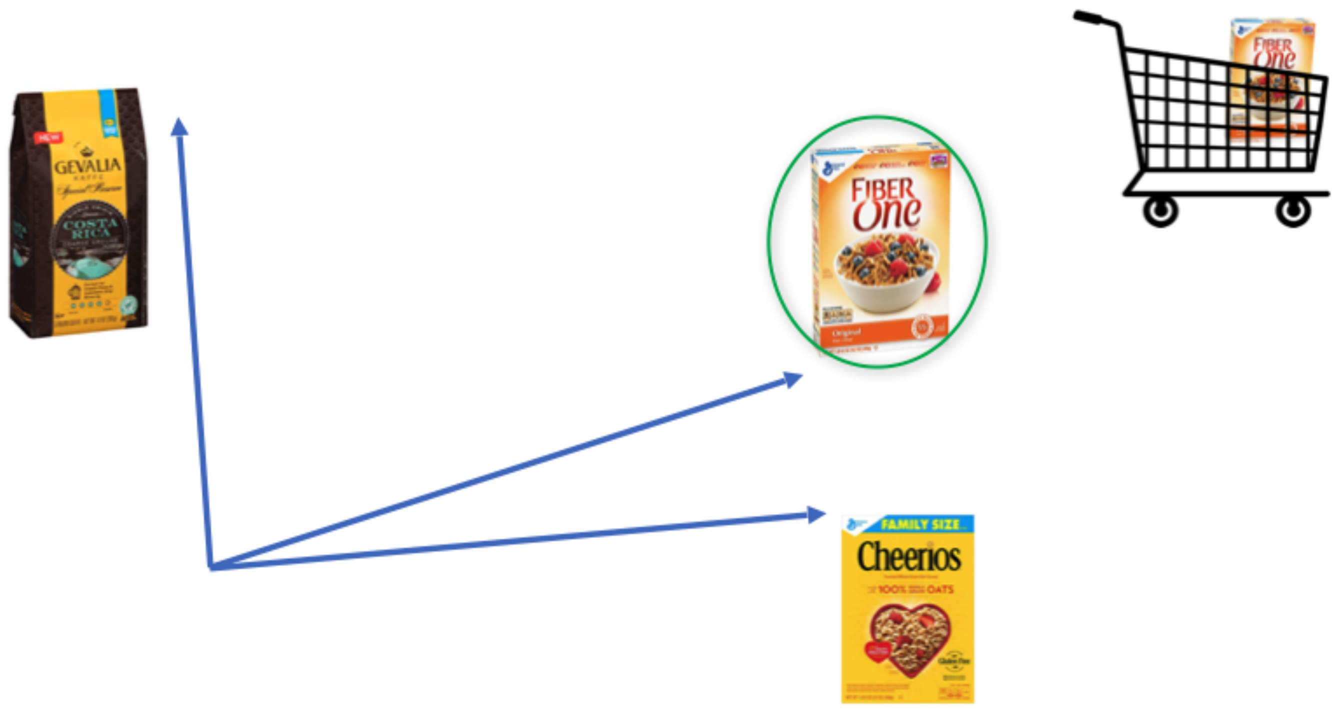
$$L = \begin{pmatrix} 1.5 & 1.9 & 1.8 & 1.4 & \cdots & 0.4 \\ 0.6 & 0.1 & 1.0 & 1.6 & \cdots & 1.9 \\ 0.6 & 1.6 & 1.6 & 1.6 & \cdots & 1.8 \\ 0.6 & 1.0 & 0.1 & 1.6 & \cdots & 0.6 \\ 0.8 & 1.4 & 1.9 & 0.8 & \cdots & 0.7 \end{pmatrix}$$

We obtain these embedding vectors from the Product2Vec service [London et al, 2017]

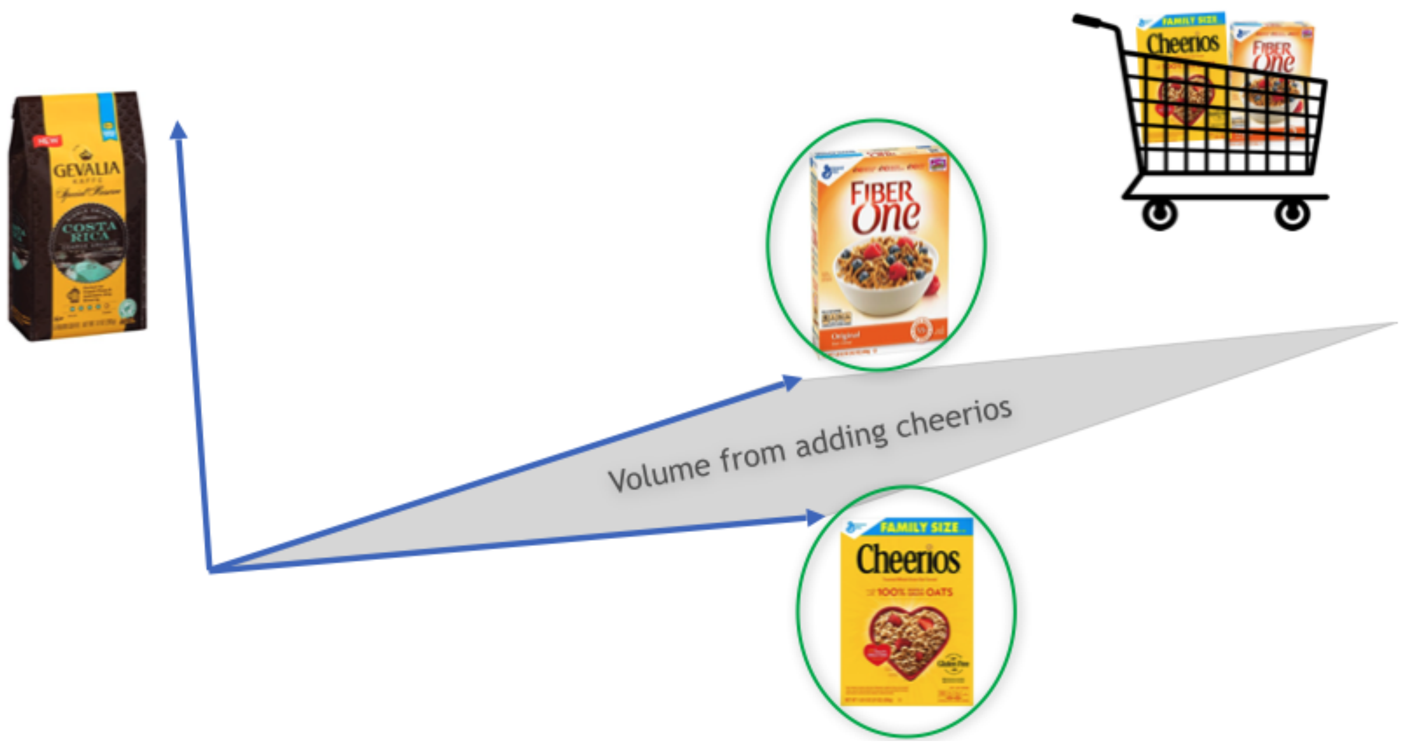
Product2Vec application



Product2Vec application



Product2Vec application



Product2Vec application

Breakout: Discuss your favorite X2Vec!

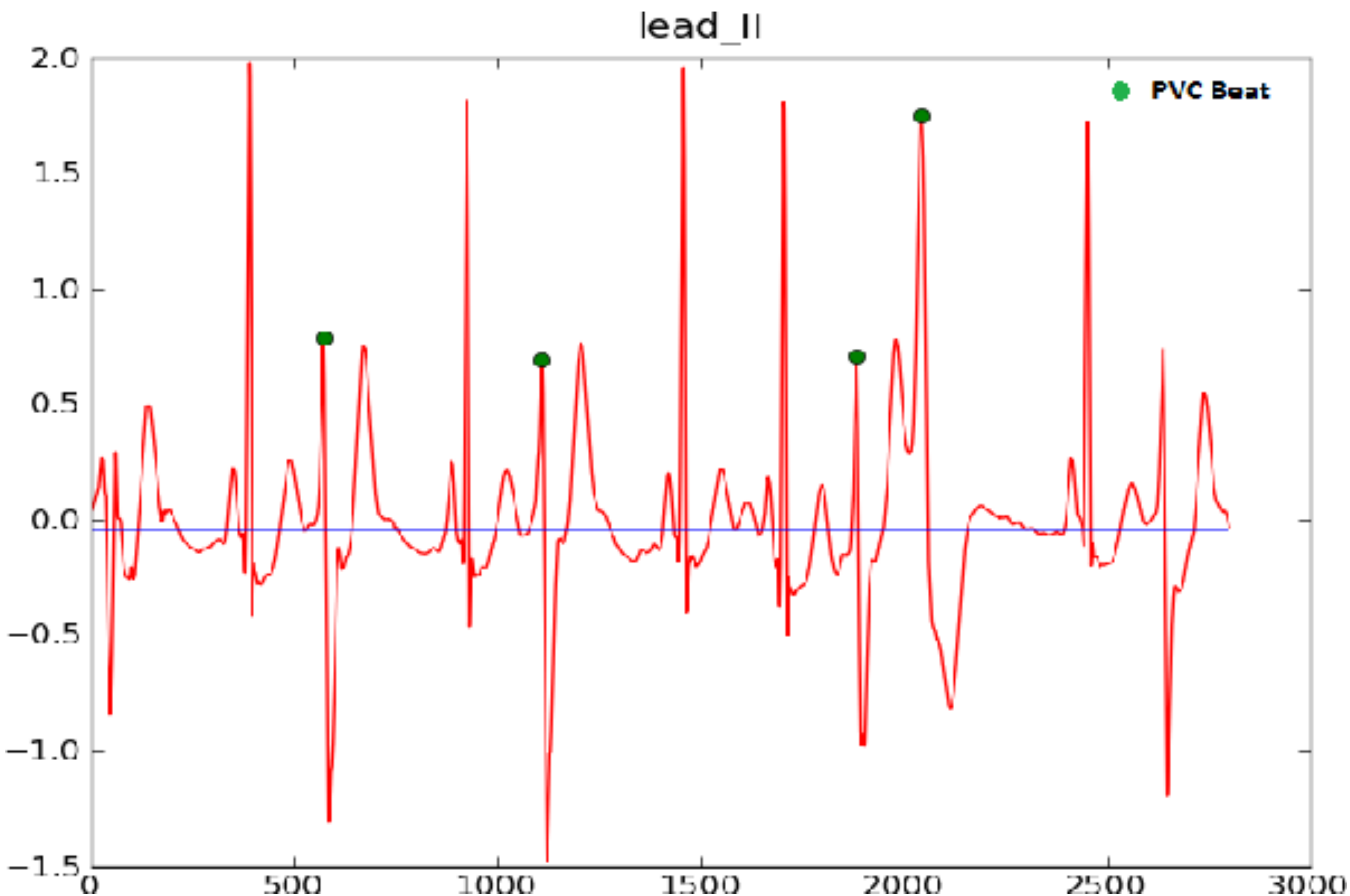
X2Vec

In your group - Discuss an application that requires machine learning. Be specific about it - Example, data, features, the type of problem (classificaiton, clustering, etc). Can you see how X2Vec would benefit your application. What would be your X in this case? How would you learn X2vec for your application? And how would you use it?

Comparison of Dimensionality Reduction Methods

Method	Utility	Pros	Cons
SVD	Low-dim embeddings	Easily	Scalability
		available	Accuracy
PCA	Same as SVD	EigenFaces	Outlier issues
Word2Vec	Semantic understanding	Non-linear	
Sentence2Vec	Comparing sentences		
Tweet2Vec	Understanding Tweets		
Product2Vec	Recommending products		
X2Vec			

Anomaly Detection: Arrythmia



Broad list of methods

Categorization

- Offline anomaly detection
- Real-time anomaly detection

Broad list of methods

Categorization

- Offline anomaly detection
- Real-time anomaly detection

Categorization

- Time-series data anomaly detection
- Regular anomaly detection

Summary

a. SVD - Bi-linear model
b. PCA application of SVD to understand variation in data
c. Application of PCA to images - Eigen faces
d. PCA can be used to compress images
e. Non-linear models more accurate and flexible
f. Word2vec uses a skip-gram non-linear model
g. Word2vec an example of context based learning (semi-supervised learning)
h. Can construct X2Vec for any X provided you have sufficient data
i. Introduction to Anomaly detection