PROJECT → Unsupervised Algorithms
→ Recommender Systems
↳ Apriori and Association Rules
↳ Matrix Factorization

"Everything is MATRIX FACTORIZATION"

- → k-means
- → GMM
- → PCA
- → SVD
- → NMF (Non-negative M.F)

Big Matrix = Smaller Matrix 1 x Smaller Matrix 2

$$\boxed{A_{n \times m} = B_{n \times d} \cdot C_{d \times m}}$$

① k-means :

↳ finds cluster centroids so that all data points are closer to the assigned centroid.

5 data points

3 clusters $(k=3)$

	c_1	c_2	c_3
x_1	1	0	0
x_2	0	0	1
x_3	0	1	0
x_4	0	0	1
x_5	1	0	0

Each row has exactly one 1.

Loss function of k-means:

$X \rightarrow$ data matrix $\rightarrow$ minimize $(X - Z.C^T)$

$$\boxed{X = Z.C^T}$$

$\rightarrow$ Special case of Matrix factorization

where, $Z \rightarrow$ cluster assignments matrix

$C \rightarrow$ centroids matrix

$$X = \begin{bmatrix} 1 & 1 \\ 1.2 & 1.1 \\ 5 & 5 \\ 4.8 & 5.1 \end{bmatrix} \quad \begin{matrix} \rightarrow c_1 \\ \\ \rightarrow c_2 \end{matrix} \qquad K = 2$$

Build matrix Z,

$c_1 \rightarrow (1,1)\ (1.2, 1.1)$

$c_2 \rightarrow (5,5),\ (4.8, 5.1)$

$$
Z = \begin{array}{c} \\ x_1 \\ x_2 \\ x_3 \\ x_4 \end{array}
\begin{array}{cc} c_1 & c_2 \\ \left[\begin{array}{c} 1 \\ 1 \\ 0 \\ 0 \end{array}\right. & \left.\begin{array}{c} 0 \\ 0 \\ 1 \\ 1 \end{array}\right] \end{array}
$$

→ Compute centroids →

$$c_1 = \left(\frac{1+1.2}{2}, \frac{1+1.1}{2} \right) = (1.1, 1.05)$$

$$c_2 = \left(\frac{5+4.8}{2}, \frac{5+5.1}{2} \right) = (4.9, 5.05)$$

$$
C = \begin{array}{c} c_1 \\ c_2 \end{array} \begin{bmatrix} 1.1 & 1.05 \\ 4.9 & 5.05 \end{bmatrix}
$$

$$X \simeq Z.C^T$$

$$= \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1.1 & 1.05 \\ 4.9 & 5.05 \end{bmatrix} = \begin{bmatrix} 1.1 & 4.9 \\ 1.05 & 5.05 \end{bmatrix}$$

$$= \begin{bmatrix} 1.1 & 1.05 \\ 1.1 & 1.05 \\ 4.9 & 5.05 \\ 4.9 & 5.05 \end{bmatrix} = \underline{\underline{Z.C}}$$

$$\begin{bmatrix} 1 & 1 \\ 1.2 & 1.1 \\ 5. & 5 \\ 4.8 & 5.1 \end{bmatrix} \simeq \begin{bmatrix} 1.1 & 1.05 \\ 1.1 & 1.05 \\ 4.9 & 5.05 \\ 4.9 & 5.05 \end{bmatrix}$$

$$\boxed{X \simeq Z.C}$$

Constraints →

→ Z always have binary values, either 0 or 1.

→ Each row sum has to be 1.

② GMM :

↳ unsupervised clustering algo. which uses probability with guassians to assign clusters.

→ Soft Clustering

3 guassians

$\odot$ → G1 G2 G3

0.6 0.3 0.1

$$\boxed{X = Z \cdot C^T}$$ → Special case of Matrix Factorisation

where, Z → cluster assignment matrix

C → guassian means.

$$Z = \begin{array}{c|ccc} & c_1 & c_2 & c_3 \\ \hline x_1 & 0.7 & 0.2 & 0.1 \\ x_2 & 0.1 & 0.1 & 0.8 \\ x_3 & 0.3 & 0.4 & 0.3 \\ x_4 & 0.1 & 0.2 & 0.7 \\ x_5 & 0.8 & 0.1 & 0.1 \end{array}$$

- Frobenius Norm: → Flattening the matrix into a vector

$$A = \begin{bmatrix} 2 & 1 \\ -3 & 4 \end{bmatrix}$$

$$\|A\|_F^2 = (2)^2 + (1)^2 + (-3)^2 + (4)^2$$
$$= 4 + 1 + 9 + 16$$
$$= 30$$

- Applications of Matrix Factorization:
  1. Recommender Systems
  2. Can be represented in Clustering, PCA, SVD.

③ Recommender Systems :

$$\boxed{A_{n \times m} = B_{n \times d} \cdot C_{d \times m}}$$

B → User embeddings
C → Item embeddings
} d – dimensions / latent factors

↓ M.F

d = Hyperparameter

d = [10 , 100]

↓ Optimal value of 'd' ?

k centroids
[2 , 10]
↓

① Elbow Curve :

x-axis → d

y-axis → Loss (Error)

→ More parameters (greater d) → Better fit → Decreased Loss

→ But after a point

↳ error convergence

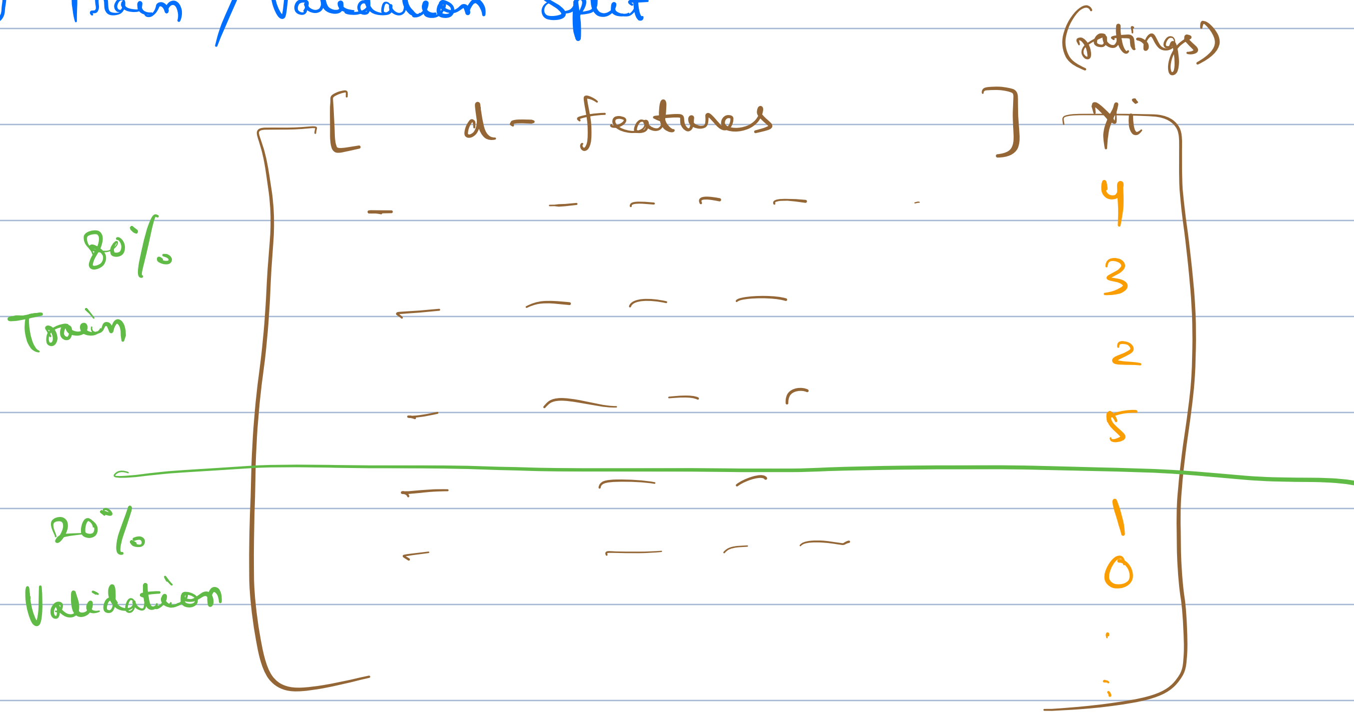
↳ it may lead to overfitting

Elbow curve approach to find optimal value of d :

↳ approximate method

↳ not perfect

② Train / Validation Split



↳ Train M.F on 80% known ratings

↳ Validate on remaining 20%

→ Pick 'd' with lowest validation error.

↳ avoids overfitting

- Interpretability of 'd':

① No real-world meaning.

② 'd' is just internal embedding dimensions.

↳ Visualize 'd'

→ Using tSNE, reduce d = 2

→ Visualize

↳ users prefering comedy

④ NMF (Non-negative MF)

→ A matrix which always have non-negative values → NMF

① Kmeans → 0, 1
② GMM → any value between 0 and 1
} NMF

③ Images → Pixel value ranges
↳ [0, 255] → NMF
↳ [0, 1] → NMF

→ But in R.S, there is no compulsion of non-negative values
↳ embeddings can be anything as long as product works.

⑤ SVD (Singular Value Decomposition)

Example:

Imagine a song with multiple instruments

- drums
- flute
- guitar

→ Can we separate a song into clean instrument tracks.

"UNPLUGGED"

SVD

↳ complicated dataset (matrix) and break into:

- basic patterns
- how strong each pattern is
- how that pattern combines to make the dataset.

FORMULA :

$$\boxed{X = U \Sigma V^T}$$

- $U \rightarrow$ left singular vectors
  ↳ shows how each data point relates to important patterns.

- $\Sigma \rightarrow$ diagonal matrix
  ↳ tells importance of each pattern

- $V^T \rightarrow$ right singular
  ↳ represents patterns

A mixed song (DESPACITO)

pure notes (patterns)

volume of each note (importance)

how they combine to form actual song

* SVD = "pattern + strength + mixing" decomposition

Why SVD is important:

1. Image Compression
2. Recommender Systems
3. Noise Reduction
4. Topic Modelling

⑥ PCA → Homework

↳ Maths

Original Matrix → Covariance Matrix (S)

↓

Eigen vectors (PCs) ← Eigen Values

$$\boxed{S = W \lambda W^T}$$

→ M.F.

→ Typical case of SVD

S = Covariance Matrix

W = Eigen vectors

λ = Eigen values

$$X = \begin{bmatrix} 2 & 2 \\ 0 & 0 \\ -2 & -2 \end{bmatrix}$$

Find $S \to ?$

Eigen values ?

Eigen vectors ?

Justify, $\boxed{S = W \lambda W^T}$

Code Implementation → M. F

→ scratch

→ use library.