TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO:

ỨNG DỤNG CỦA PHÂN RÃ SVD
(SINGULAR VALUE DECOMPOSITON)

Sinh viên thực hiện**:**
Tên sinh viên: Ngô Quốc Anh **Mã SV: 102220002**

Đà Nẵng, 09/2024

MỤC LỤC

MỤC LỤC i
MỞ ĐẦU i
Chương 1. TỔNG QUÁT SVD 1
1.1. Giới thiệu 1
1.2. Các bước tính SVD 1
Chương 2. HAI DẠNG VÀ MỘT SỐ ỨNG DỤNG CỦA SVD 2
2.1. Compact SVD và Truncated SVD 2
2.2. Phân tích thành phần chính (PCA) – giảm chiều dữ liệu 3
2.3. Hệ thống Recommendation Systems (RS) 5
2.4. Giải hệ phương trình tuyến tính 6
KẾT LUẬN 9

MỞ ĐẦU

Trong học máy và khoa học dữ liệu hiện đại, việc phân tích một dữ liệu lớn, phức tạp là điều không thể tránh khỏi và ngày càng quan trọng. Trong đó có một phương pháp để giải quyết vấn đề này, đó là phân rã giá trị suy biến (Singular Value Decompositon) – gọi tắt là SVD.

Có nhiều phương pháp để phân rã như: phân rã Cholesky, phân rã theo giá trị riêng và vector riêng,… Nhưng các phương pháp trên chủ yếu dùng trong tính toán, không được sử dụng nhiều trong học máy. Trong thực tế, ma trận không vuông tồn tại rất nhiều, nên việc giải quyết các ma trận không vuông là phương pháp phân rã giá trị suy biến SVD.

Chương 1. TỔNG QUÁT SVD

1.1. Giới thiệu

SVD là phương pháp giúp phân rã bất cứ ma trận nào thành tích của 3 ma trận với tính chất đặc biệt.

Giả sử ta có ma trận A với kích thước *m x n*. SVD của ma trận A có dạng:

$$\boldsymbol{A_{mxn} = U\,D\,V^T}$$

Trong đó:

- U: ma trận trực giao kích thước *m x m*, chứa các vector riêng của của ma trận AA^T (còn gọi là ma trận vector riêng trái của A).
- V: ma trận trực giao kích thước *n x n*, chứa chứa các vector riêng của của ma trận A^TA (còn gọi là ma trận vector riêng phải của A).
- D: ma trận đường chéo kích thước *m x n*, chứa các singular values của ma trận A. Các giá trị này được sắp xếp giảm dần theo giá trị tuyệt đối. Ma trận D có dạng: $\begin{pmatrix} \delta 1 & 0 & 0 & 0 \\ 0 & ... & 0 & 0 \\ 0 & 0 & \delta r & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$.

1.2. Các bước tính SVD

1. Tính các vector riêng và giá trị riêng của ma trận AA^T và A^TA, từ đó xác định ma trận U và V.
2. Các giá trị $\delta i = \sqrt{\lambda i}$ của ma trận D.

Chương 2. HAI DẠNG VÀ MỘT SỐ ỨNG DỤNG CỦA SVD

2.1. Compact SVD và Truncated SVD

Compact SVD là một phiên bản tối ưu hóa của phân tích giá trị suy biến (SVD), được sử dụng khi bạn chỉ quan tâm đến một số lượng hạn chế các thành phần chính của ma trận dữ liệu. Nó giúp tiết kiệm bộ nhớ và tính toán, đặc biệt là khi làm việc với ma trận lớn.

Giả sử có một ma trận A kích thước *m x n* (với m ≥ n), compact SVD chỉ tính toán các thành phần chính quan trọng.

- Phân tích giá trị suy biến SVD:

$$\boldsymbol{A_{mxn} = U_{mxm}\, D_{mxn}\, V_{nxn}^T = [u_1, u_2, \ldots, u_m] * \begin{pmatrix} \delta 1 & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & 0 \\ 0 & \cdots & \delta r & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} * [v_1^T, v_2^T, \ldots, v_n^T]}$$

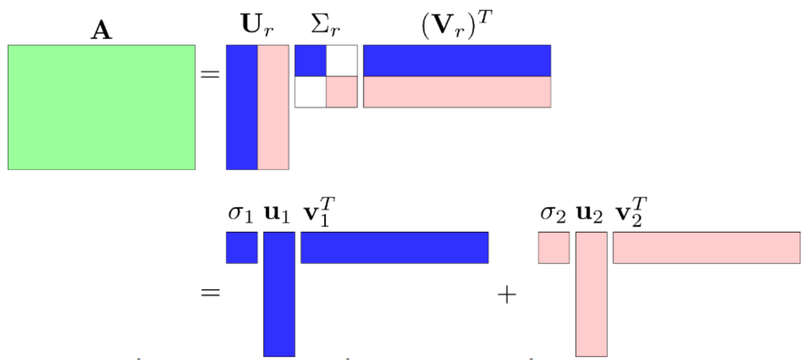
- Nhân các ma trận, ta có:

$$\boldsymbol{A_{mxn} = \sum_{i=1}^{r} \delta_i u_i v_i^T}$$

Rõ ràng trong cách biểu diễn này, ma trận A chỉ phụ thuộc vào r cột đầu tiên của U, V và r giá trị khác 0 trên đường chéo của ma trận D. Vậy compact SVD có dạng:

$$\boldsymbol{A = U_r\, D_r\, V_r^T}$$

D_r là ma trận con được tạo bởi r hàng đầu tiên và r cột đầu tiên của D. Nếu ma trận X có rank nhỏ hơn rất nhiều so với số hàng và số cột r ≪ m , n , ta sẽ được lợi nhiều về việc lưu trữ.



Nếu từ đầu phải sử dụng ma trận m hàng, n cột để lưu trữ ma trận A, thì đối với compact SVD, ta chỉ cần sử dụng n + m + 1.

Truncated SVD thì loại bỏ cả các singular values nhỏ. Có nghĩa là chủ động chọn một số lượng các thành phần chính được cho là quan trọng nhất (thường là các singular values lớn nhất).

Compact SVD giữ lại tất cả các giá trị singular values khác 0, trong khi Truncated SVD chỉ giữ lại một số lượng nhỏ các singular values quan trọng nhất.

Về mặt toán học: $\boldsymbol{X_k = U_k\, D_k\, V_k^T}$

Trong đó $k < r$, và k là số lượng thành phần chính muốn giữ lại (thường được chọn sao cho phần lớn biến thiên của dữ liệu được bảo toàn).

2.2. Phân tích thành phần chính (PCA) – giảm chiều dữ liệu

Principal Component Analysis (PCA) một kỹ thuật phân tích dữ liệu phổ biến được sử dụng để giảm chiều dữ liệu, nén dữ liệu và trích xuất các thành phần chính quan trọng từ dữ liệu. PCA giúp biến đổi dữ liệu từ không gian chiều D xuống không gian chiều k (k < D) và phải giữ được một số thuộc tính có ý nghĩa của dữ liệu gốc.

Trong PCA, mục tiêu của chúng ta là tìm các hướng mới trong không gian của các biến để giảm chiều dữ liệu. Trước khi PCA, cần chuẩn hóa ma trận. Điều này là do PCA là một thuật toán dựa trên phương sai, và việc chuẩn hóa sẽ đảm bảo rằng tất cả các thuộc tính (tính năng) trong dữ liệu có tầm quan trọng tương đương khi tính toán phương sai.

Giả sử có một tập dữ liệu X_{mxn}, với mỗi đặc trưng (cột) x_j, chuẩn hóa Z-Scores:

$$z_j = \frac{xj - \mu j}{\sigma j}$$

Trong đó: x_j là giá trị của một đặc trưng (cột) thứ j.

μj là trung bình của đặc trưng j.

σj là độ lệch chuẩn của đặc trưng j.

Tính trung bình và độ lệch chuẩn cho từng cột:

$$\boldsymbol{\mu j = \frac{1}{m}\sum_{i=1}^{m} xij}$$

$$\boldsymbol{\sigma j = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(xij - \mu j)^2}}$$

Chuẩn hóa từng giá trị:

$$z_j = \frac{xij - \mu j}{\sigma j}$$

Khi nhân ma trận X với V, chúng ta đang thực hiện phép chiếu các điểm dữ liệu trong không gian gốc X lên không gian mới được định nghĩa bởi các vector riêng trong V.

Ma trận thành phần chính:

$$\boldsymbol{T = X_{normalized}V}$$

Trong đó V chứa các trục v_1, v_2,…, v_n, và T là tọa độ mới của dữ liệu trong không gian này. V là ma trận vector riêng của ma trận hiệp phương sai, thể hiện các hướng chính của dữ liệu trong không gian đặc trưng.

(T là ma trận dữ liệu sau khi chuyển đổi qua PCA, tức là dữ liệu được biểu diễn theo các trục chính, giúp nén và giảm chiều nhưng vẫn giữ được những yếu tố chính của dữ liệu gốc)

Mỗi hàng trong T sẽ cho ta tọa độ của một điểm dữ liệu tương ứng trong không gian mới. Tọa độ này cho biết mức độ tương quan của điểm dữ liệu đó với từng thành phần chính.

Mà $X = UDV^T$ nên suy ra ta có:

$$\boldsymbol{T = UDV^TV}$$

$$\Rightarrow \boldsymbol{T = UD}$$

Tỷ lệ biến thiên = $\frac{\sum_{k=1}^{r} \lambda k}{\sum_{k=1}^{n} \lambda k}$

Trong đó: λk là bình phương của các singular values σk trong ma trận D

r là số lượng thành phần chính giữ lại

n là tổng số thành phần chính

Công thức này cho biết tỷ lệ biến thiên của dữ liệu được nắm giữ bởi r thành phần chính so với toàn bộ biến thiên của dữ liệu. Ví dụ, chỉ giữ lại 2 principal components đầu tiên và chúng chiếm 90% tổng biến thiên, điều này có nghĩa là phần lớn thông tin của dữ liệu đã được giữ lại trong 2 thành phần chính này.

Giả sử với ma trận X = $\begin{pmatrix} 2 & 1 \\ 1 & 2 \\ 1 & -1 \end{pmatrix}$, chuẩn hóa Z-Scores ta được:

$X_{normalized} = \begin{pmatrix} 1.417 & 0.266 \\ -0.7 & 1.068 \\ -0.7 & -1.335 \end{pmatrix}$

Phân rã $X_{normalized}$ ta thu được:

$U = \begin{pmatrix} -0.6301 & 0.5243 & 0.5727 \\ -0.1396 & -0.8020 & 0.5806 \\ 0.7638 & 0.2858 & 0.5786 \end{pmatrix}$, $D = \begin{pmatrix} 1.8853 & 0 \\ 0 & 1.5578 \\ 0 & 0 \end{pmatrix}$

Suy ra, ta có $T = UD = \begin{pmatrix} -0.6301 & 0.5243 & 0.5727 \\ -0.1396 & -0.8020 & 0.5806 \\ 0.7638 & 0.2858 & 0.5786 \end{pmatrix} * \begin{pmatrix} 1.8853 & 0 \\ 0 & 1.5578 \\ 0 & 0 \end{pmatrix} =$

$$\begin{pmatrix} -1.188 & 0.8168 \\ -0.2632 & -1.25 \\ 1.44 & 0.4452 \end{pmatrix}$$

Tỷ lệ biến thiên với cột 1: $\alpha 1 = \frac{\sum_{k=1}^{r} \lambda k}{\sum_{k=1}^{n} \lambda k} = 0.6$

Tỷ lệ biến thiên với cột 2: $\alpha 2 = \frac{\sum_{k=1}^{r} \lambda k}{\sum_{k=1}^{n} \lambda k} = 0.4$

Vậy chỉ với 1st principal component, nó đã chiếm 60% biến thiên nên ta có thể giảm chiều ma trận.

$$T = \begin{pmatrix} -1.188 \\ -0.2632 \\ 1.44 \end{pmatrix}$$

2.3. Hệ thống Recommendation Systems (RS)

Hệ thống gợi ý thường dựa vào một ma trận người dùng - sản phẩm, trong đó mỗi hàng đại diện cho một người dùng và mỗi cột đại diện cho một sản phẩm. Giá trị tại mỗi ô của ma trận thể hiện đánh giá của người dùng cho sản phẩm tương ứng. Tuy nhiên, ma trận này thường rất thưa (nhiều ô trống do không có đánh giá), khiến việc dự đoán các đánh giá còn thiếu trở nên khó khăn.

Nếu ta có ma trận X có kích thước m x n là cách biểu diễn mối quan hệ giữa người dùng và sản phẩm, với m là số người dùng và n là số sản phẩm. Mỗi phần tử X_{ij} của ma trận tại hàng i và cột j biểu thị đánh giá của người dùng i đối với sản phẩm j. Nếu người dùng chưa đánh giá sản phẩm, phần tử đó thường để trống hoặc có giá trị mặc định (0 hoặc -∞).

Vì vậy nên ta phải xây dựng ma trận $\hat{X}$ đầy đủ dữ liệu để có thể lấp đầy ma trận X. Ma trận $\hat{X}$ được xây dựng dựa trên 3 ma trận phân rã SVD từ X.

$$\begin{array}{cccc} \hat{X} & \mathrm{U} & \mathrm{S} & \mathrm{V^T} \end{array}$$

$$\underbrace{\begin{pmatrix} x11 & x12 & \ldots & x1n \\ x21 & x22 & \ldots & x2n \\ \vdots & \vdots & \ldots & \ldots \\ xm1 & xm2 & \ldots & xmn \end{pmatrix}}_{m\,x\,n} \approx \underbrace{\begin{pmatrix} u11 & \cdots & u1r \\ \vdots & \ddots & \vdots \\ um1 & \cdots & umr \end{pmatrix}}_{m\,x\,r} \underbrace{\begin{pmatrix} s11 & 0 & \ldots \\ 0 & \ddots & \vdots \\ \vdots & \cdots & srr \end{pmatrix}}_{r\,x\,r} \underbrace{\begin{pmatrix} v11 & \cdots & u1n \\ \vdots & \ddots & \vdots \\ ur1 & \cdots & urn \end{pmatrix}}_{r\,x\,n}$$

Để tìm được ma trận $\hat{X}$, ta thực hiện n vòng lặp, mỗi bước lặp thực hiện:

Thông qua X ta có 3 ma trận:

$U_0, S_0, V_0^T \rightarrow \hat{X}_0$, $\varepsilon_0 = \|X - \hat{X}0\|$

Nếu độ lệch còn lớn, tiếp tục thực hiện thông qua U_0, S_0, V_0^T:

$U_1, S_1, V_1^T \rightarrow \hat{X}_1$, $\varepsilon_1 = \|X - \hat{X}1\|$, $(\varepsilon_1 < \varepsilon_0)$

…

…

Cứ tiếp tục n bước lặp, tại bước k nào đó, ta tìm được:

$U_k, S_k, V_k^T \rightarrow \hat{X}_k$, $\varepsilon_k = \|X - \hat{X}k\| \approx 0$, $(\varepsilon_k < \varepsilon_{k-1})$

⇨ $X \equiv \hat{X}_k = \hat{X}$

Vậy ta thu được $\hat{X}$ có đầy đủ giá trị. Như vậy ta tìm được X_{new} có độ lệch không quá lớn so với X với đầy đủ giá trị.

Như vậy RS giúp tạo ra trải nghiệm được tùy chỉnh riêng cho từng người dùng. Thay vì hiển thị nội dung chung cho tất cả mọi người, hệ thống gợi ý hiển thị nội dung phù hợp với sở thích và hành vi của mỗi người dùng.

2.4. Giải hệ phương trình tuyến tính

SVD cung cấp một cách tiếp cận số học ổn định để xử lý các hệ phương trình tuyến tính, đặc biệt khi ma trận A không vuông hoặc không có hạng đầy đủ. SVD phân rã ma trận hệ số A thành tích của ba ma trận khác, cho phép chúng ta xử lý và giải hệ phương trình một cách hiệu quả.

Giả sử có hệ phương trình

$$\boldsymbol{Ax = B}$$

Trong đó: A là ma trận hệ số (kích thước *m x n*)

x là vector ẩn (kích thước *n x 1*)

B là vector kết quả (kích thước *m x 1*)

Phân rã SVD ma trận A thành tích của ba ma trận:

$$\boldsymbol{A = UDV^T}$$

Nhân cả hai vế với U^T ta có:

$$\boldsymbol{DV^Tx = U^TB}$$

Đặt $\boldsymbol{y = V^Tx}$, ta có:

$$\boldsymbol{Dy = U^TB}$$

Tiếp tục, ta giải y từ hệ phương trình trên bằng cách chia từng thành phần trên đường chéo của D (bỏ qua các giá trị bằng 0 nếu có)

Sau khi có y, ta tính x từ:

$$\boldsymbol{x=Vy}$$

- Ví dụ với $A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{pmatrix}$, $B = \begin{pmatrix} 7 \\ 8 \\ 9 \end{pmatrix}$

Phân rã SVD ma trận $A = UDV^T$ thu được:

$$U = \begin{pmatrix} -0.2298 & 0.8834 & 0.4082 \\ -0.5247 & 0.2408 & -0.8165 \\ -0.8196 & -0.4019 & 0.4082 \end{pmatrix}$$

$$D = \begin{pmatrix} 9.5255 & 0 \\ 0 & 0.5143 \\ 0 & 0 \end{pmatrix}$$

$$V^T = \begin{pmatrix} -0.6196 & -0.7849 \\ -0.7849 & 0.6196 \end{pmatrix}$$

Tính U^TB:

$$U^TB = \begin{pmatrix} -0.2298 & -0.5247 & -0.8196 \\ 0.8834 & 0.2408 & -0.4019 \\ 0.4082 & -0.8165 & 0.4082 \end{pmatrix} \begin{pmatrix} 7 \\ 8 \\ 9 \end{pmatrix} = \begin{pmatrix} -13.1826 \\ 4.4931 \\ -0.0008 \end{pmatrix}$$

Tính y:

$$Dy = U^TB$$

$$\Leftrightarrow \begin{pmatrix} 9.5255 & 0 \\ 0 & 0.5143 \\ 0 & 0 \end{pmatrix} y = \begin{pmatrix} -13.1826 \\ 4.4931 \\ -0.0000 \end{pmatrix}$$

$$\Leftrightarrow y = \begin{pmatrix} -1.3840 \\ 8.7363 \end{pmatrix}$$

Tính x:

$$\begin{pmatrix} -0.6196 & -0.7849 \\ -0.7849 & 0.6196 \end{pmatrix} x = \begin{pmatrix} -1.3840 \\ 8.7363 \end{pmatrix}$$

$$x = \begin{pmatrix} -6 \\ 6.5 \end{pmatrix}$$

Vậy phương trình có 2 nghiệm $x_1 = -6$, $x_2 = 6.5$.

KẾT LUẬN

SVD là một công cụ cơ bản nhưng vô cùng hữu ích trong nhiều lĩnh vực. Với sự phát triển của máy học và khoa học dữ liệu, vai trò của SVD càng trở nên quan trọng hơn. Việc hiểu rõ về SVD và các ứng dụng của nó sẽ giúp bạn giải quyết nhiều bài toán thực tế một cách hiệu quả.