〈국문초록〉

데이터의 행렬화를 이용한 효율적인 추천 시스템에 대한 연구

연구자 : 류수현(2학년, danny906@naver.com)
박지호(2학년, hy7720@naver.com)
정세화(2학년, newton0512@naver.com)

책임지도자 : 전영철(KAIST 부설 한국과학영재학교, ycjeon@kaist.ac.kr)

요약문

본 연구에서는 개개인의 특성을 중요시하는 추천 시스템을 구상하였다. 사용자가 제시한 평가들을 바탕으로 상품의 평가기준들 중 가장 중요한 n개의 요소들에 해당되는 값을 행렬화해 벡터 $e_1, e_2, \cdots, e_n$을 만들 수 있다. 그 후, 각각의 벡터들에게 주어지는 계수들을 Least Square Method와 Singular Value Decomposition(SVD)를 이용하여 예측한다. 이 계수는 개인마다 다르게 주어지며 최적화된 값이라고 할 수 있다.

주제어 : 추천 시스템, 데이터의 행렬화, Least Square Method, SVD

Ⅰ. 서론

최근에 추천 시스템(recommender system)이 화두가 되고 있다. 2006년도에는 Netflix Prize에서 이용자들이 별점을 부여한 데이터를 바탕으로 추천 시스템의 정확도를 높이는 대회가 개최되기도 하였다. 추천 시스템의 가장 중요한 요소는 item 간 혹은 user 간의 관계를 확인함으로써 관계를 찾아보는 것이다. 관계를 정확히 파악할수록 더 적합한 결과를 예측할 수 있다.

본 연구에서는 많은 item들 중에서도 영화를 추천하는 시스템을 구상하기로 했다. 사람들이 영화를 평가하는 기준은 아주 다양하다. 감독, 배우, 작품성, 인지도, 장르 등등 많은 요소들이 있고 이 요소들이 모든 사람들에게 각각 다른 비중으로 작용한다. 따라서 각각의 사람에 대해 다른 가중치를 주어야 할 필요성이 있다. 따라서 개개인이 중요시하는 요소에 맞게 영화를 추천해줄 수 있는 시스템을 구상하였다.

작품 간, 사용자 간의 관계를 확률로 나타낸다고 생각할 수 있다. 그러나 많은 데이터들을 수용할 수 있고 비교적 간단한 연산을 위해 선형대수학적 행렬화와 연산을 이용하기로 결정하였다.

Ⅱ. 이론적 배경

1. Gram-Schmidt Orthogonalization

Definition1. 임의의 basis를 orthogonal 하도록 변환하는 과정

대부분의 경우 orthogonal basis 는 우리가 찾을 수 있는 가장 좋은 basis 이다. 그러나 많은 경우에는 orthogonal basis 가 직접적으로 주어지지 않기 때문에 임의의 basis를 orthogonal하도록 만들어 주는 과정이 필요하다. 이것이 gram-schmidt process이다.

$\{w_1, w_2, \dots, w_n\}$ 을 inner product space V 의 basis 일 때, $v_1 = w_1$로 두고
span$\{w_1, w_2, \dots, w_n\}$ = span$\{v_1, v_2, \cdots, v_n\}$이고 orthogonal 한 $\{v_1, v_2, \cdots, v_n\}$을 찾자.
우선, v_1에 수직인 벡터 v_2를 찾자.
만약 $< v_1, w_2 >= 0$ 이면 $v_2 = w_2$로 둔다.

만약 $< v_1, w_2 > \neq 0$ 이면, $w_2 = a_1 v_1 + w_2$이고 $< v_2, v_1 > = 0$인 $a_1 \in R$을 찾는다.

$$a_1 = - \frac{< w_2, v_1 >}{< v_1, v_1 >},\ v_2 = - \frac{< w_2, v_1 >}{< v_1, v_1 >} v_1 + w_2$$

그 다음, $< v_3, v_2 > = < v_3, v_1 > = 0$을 만족시키는 $v_3 = b_1 w_1 + b_2 w_2 + w_3$ 를 찾는다.
앞서 구한 v_2를 대입하면

$$v_3 = - \frac{< w_3, v_1 >}{< v_1, v_1 >} v_1 - \frac{< w_3, v_2 >}{< v_2, v_2 >} v_2 + w_3$$

같은 방법으로

$$v_k = - \sum_{i=1}^{k-1} \left(\frac{< w_k, v_1 >}{< v_1, v_1 >} \right) v_i + w_k$$

최종적으로 얻어낸 $\{v_1, v_2, \ldots v_n\}$는 vector space V의 orthogonal basis 가 된다.

2. Projection

Definition2. $P^2 = P$를 만족하는 square matrix P 를 **projector** 라 한다.

Definition3. 만약 P 가 projector이면, $I - P$ 도 projector가 된다는 사실을 쉽게 알 수 있다.
이러한 $I - P$를 **complementary projector**라 한다.

complementary projector에서 아래와 같은 몇 가지 성질을 얻어낼 수 있다.

① $R(I-P) = N(P)$

② $N(I-P) \cap N(P) = 0$

pf〉 1) $R(I-P) = N(P)$ 임을 보이는 문제는

$R(I-P) \subseteq N(P)$ 이고 $R(I-P) \supseteq N(P)$ 임을 보이는 문제와 같다.

$\forall v \in R(I-P),\ \exists x\ s.t.\ v = (I-P)x \Rightarrow Pv = P(I-P)x = 0 \Rightarrow v \in N(P)$

마찬가지로

$\forall w \in N(P), Pw = 0 \Rightarrow (I-(I-P))w = 0 \Rightarrow w = (I-P)w \Rightarrow w \in R(I-P)$

따라서 $R(I-P) = N(P)$ 이다.

2) $\exists v \in N(I-P) \cap N(P) \Rightarrow (I-P)v = 0, Pv = 0 \Rightarrow v = 0$

따라서 $N(I-P) \cap N(P) = \{0\}$

Definition4. Orthogonal projector는 range 와 null space가 orthogonal subspace인 projection을 말한다.

LEM1. $m \times n$ real matrix A, $x \in \mathbb{R}^n, y \in \mathbb{R}^m$에 대해 $\langle Ax, y \rangle = \langle x, A^T y \rangle$ 이다.

pf〉 어떤 z가 $\langle Ax, y \rangle = \langle x, z \rangle$을 만족한다고 하자.

A의 i번째 행 j번째 열에 위치한 원소를 a_{ij}라 하면

$$\langle Ax, y \rangle = \left\langle \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ & \vdots & \\ a_{m1} & \cdots & a_{mn} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}, \begin{bmatrix} y_1 \\ \vdots \\ y_m \end{bmatrix} \right\rangle = \left\langle \begin{bmatrix} \sum_{i=1}^{n} a_{1i} x_i \\ \vdots \\ \sum_{i=1}^{n} a_{mi} x_i \end{bmatrix}, \begin{bmatrix} y_1 \\ \vdots \\ y_m \end{bmatrix} \right\rangle$$

$$\sum_{j=1}^{m} (\sum_{i=1}^{n} a_{ji} x_i) y_j = \sum_{i=1}^{n} x_i (\sum_{j=1}^{m} a_{ji} y_j)$$

$$z_i = \sum_{j=1}^{m} a_{ji} y_j \Rightarrow z = A^T y$$

따라서 $\langle Ax, y \rangle = \langle x, A^T y \rangle$가 성립한다.

Theorem1. Let P be a projector with $P = P^*$.

Then P is an orthogonal projector.

pf〉 $\forall v \in R(P), u \in N(P),\ v = Px, u = (I-P)x$ for *some* x, y

$< v, u > = < Px, (I-P)y > = < x, P^*(I-P)y > = < x, P(I-P)y > = 0$

따라서 P는 orthogonal projection 이다.

$Q^*Q = I$ 를 만족하는 m×n matrix Q 를 생각해보자 우리는 QQ^*가 orthogonal projector 인 것을 쉽게 확인 할 수 있다. $P = QQ^*$이라 하자. 그러면 앞선 보조정리를 통해 $R(P) = N(I-P)$ 이 성립한다. 만약 $\mathbb{X} \in R(P)$이면 $\mathbb{X} \in N(I-P)$ 이므로 $(I-QQ^*)\mathbb{X} = 0 \Leftrightarrow \mathbb{X} = QQ^*\mathbb{X}$

Q의 j번째 column을 q_j라 하자.

$$QQ^*\mathbb{X} = [q_1 q_2 \cdots q_n]\begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ q_n \end{bmatrix}\mathbb{X} = [q_1 q_2 \cdots q_n]\begin{bmatrix} < q_1, \mathbb{X} > \\ < q_2, \mathbb{X} > \\ \vdots \\ < q_n, \mathbb{X} > \end{bmatrix}$$

$$\mathbb{X} = q_1 < q_1, \mathbb{X} > + \ldots + q_n < q_n, \mathbb{X} > \in span\{q_1, q_2, \ldots q_n\} = R(Q)$$

임의의 basis를 이용한 projection을 해보자.

$\mathbb{C}^m$의 subspace W에 대하여 W의 임의의 basis를 $\{a_1, a_2, \ldots a_n\}$이라고 하고 i번째 column이 a_i인 $m \times n$ matrix를 A라 하자. vector v를 $R(A)$에 대해 project한 vector를 y라 하면 $y \in W$ 이므로 $\exists\, x_i 1 \leqq i \leqq n \; s.t \; y = \sum_{i=1}^{n} x_i a_i$

$$a_j^*(y - v) = 0, \forall \; 1 \; \leq \; \mathrm{j} \; \leq \; \mathrm{n}$$

$$\Leftrightarrow a_j^*(\sum_{i=1}^{n} x_i a_i - v) = 0$$

$$\Leftrightarrow a_j^*(A\mathbb{X} - v) = 0 \text{ for } \forall j$$

$$\Leftrightarrow A^*(A\mathbb{X} - v) = 0 \Leftrightarrow A^*A\mathbb{X} = A^*v$$

한편, $A^*A\mathbb{X} = 0 \Rightarrow \mathbb{X}^*A^*A\mathbb{X} = 0 \Rightarrow \| A\mathbb{X} \|_2^2 = 0 \Rightarrow A\mathbb{X} = 0 \Rightarrow \mathbb{X} = 0$ 이므로 A^*A는 invertible하다.

따라서

$$\mathbb{X} = (A^*A)^{-1}A^*v \Rightarrow \; y = A(A^*A)^{-1}A^*v$$

3. Singular Value Decomposition (SVD)

3.1. Symmetric Mat. is orthogonally diagonalizable

Definition5. $A = A^T$인 matrix A를 **symmetric matrix**라 하고,

$U^{-1} = U^T$인 matrix U를 **orthogonal matrix**라고 한다.

Definition6. $n \times n$ matrix A가 orthogonal matrix U와 diagonal matrix D에 대해

$A = UDU^{-1}$로 표현된다면 A를 **orthogonally diagonalizable**하다고 부른다.

Theorem2. 만약 A가 orthogonally diagonalizable 하다면 A는 symmetric하다.

pf〉 A가 orthogonally diagonalizable하다면 orthogonal matrix인 U와
diagonal matrix D에 대해 $A = UDU^{-1}$로 표현되고

$$A^T = (UDU^T)^T = U^{TT}D^TU^T = UDU^T = UDU^{-1} = A$$

이므로 A는 symmetric하다.

Theorem3. $n \times n$ matrix A가 symmetric하다.

$\Leftrightarrow \mathbb{R}^n$ 의 vectors x와 y에 대해 $Ax \cdot y = x \cdot Ay$이다.

pf〉 ($\Rightarrow$)

Let x,y be in $\mathbb{R}^n$. For any $n \times n$ matrix A

$Ax \cdot y = (Ax)^T y = x^T A^T y = x \cdot A^T y$

만약 A가 symmetric matrix라면 $A = A^T$이므로 $Ax \cdot y = x \cdot Ay$

($\Leftarrow$)

$Ax \cdot y = x \cdot Ay$ for all vectors x and y in $\mathbb{R}^n$ 가 성립한다 하고. 그리고

$a_1, a_2, ..., a_n$을 A의 columns라 하자. 그러면 $1 \le i,j \le n$인 모든 i,j에 대해

$Ae_i \cdot e_j = a_i \cdot e_j = a_{ji}$ 가 되고 따라서 A는 symmetric matrix가 된다.

$\Updownarrow$

$e_i \cdot Ae_j = e_i \cdot a_j = a_{ij}$

LEM2. 모든 complex vector z에 대해서 $q = \bar{z}^T Az$는 real이다.

pf〉 $z = \begin{bmatrix} z_1 \\ . \\ . \\ . \\ z_n \end{bmatrix} \in \mathbb{C}^n$ 인 complex vector를 잡고 $q = \bar{z}^T Az$인 q에 대해 생각해보자.

$$q = \bar{z}^T Az = \bar{z} \cdot Az = A\bar{z} \cdot z = z \cdot A\bar{z} = z^T A\bar{z} = z^T \overline{A}\bar{z} = \bar{q}$$

가 되므로 q는 real이다.

Theorem4. A가 $n \times n$ real symmetric matrix이면, n real eigenvalues를 가진다.
또한, 각각의 eigenvalue에 대해서 real eigenvector를 찾을 수 있다.

pf〉 A가 $n \times n$ matrix이므로 A는 n개의 complex eigenvalues를 가진다.

λ_i을 eigenvalue 중 하나라 하고 이에 대응되는 0이 아닌 eigenvector 중 하나를 $z = \begin{bmatrix} z_1 \\ . \\ . \\ . \\ z_n \end{bmatrix}$ 라 하자.

그러면 $\bar{z}^T Az = \bar{z}^T \lambda_i z = \lambda_i \bar{z}^T z = \lambda_i \bar{z} \bullet z = \lambda_i (|z_1|^2 + ... + |z_n|^2)$이고 $\bar{z}^T Az$는 LEM2에 의해 실수, $(|z_1|^2 + ... + |z_n|^2)$ 역시 실수이므로 λ_i가 실수임을 알 수 있다.

λ_i가 real이므로 A-$\lambda_i I$ 는 real matrix이고 $\det(A - \lambda_i I) = 0$이므로 $(A - \lambda_i I)x = 0$은 real nonzero solution을 가지기 때문에 real eigenvector가 존재한다.

Theorem5. $n \times n$ matrix A가 orthogonally diagonalizable하다

$\Leftrightarrow$ A is symmetric하다

pf〉 ($\Rightarrow$) 앞의 Theorem2 에 의해서 증명되었다.

($\Leftarrow$)

n에 대해서 귀납법을 써보자.

n=1인 경우, $A = [a]$라고 할 때 $A = [1][a][1]$로 성립한다.

가정 : $k \geq 1$에 대하여 $k \times k$ symmetric matrix는 orthogonally diagonalizable하다.

이제 $(k+1) \times (k+1)$ symmetric matrix애 대해 생각해보자.

Theorem4에 따라 real eigenvalue λ_i과 real eigenvector v_i를 찾고 각각의 norm으로 나누어 주어 v_i를 unit eigenvector로 만들자. Gram Schmidt process를 이용하면 orthonormal basis $v_1 ... v_n$을 얻을 수 있다.

P를 v_i를 column으로 가지는 행렬이라 하자. $P = [v_1 ... v_n]$

P는 orthogonal이므로 $P^{-1} = P^T$이다.

$P^{-1}AP$를 살펴보면

$$(P^{-1}AP)^T = (P^T AP)^T = P^T A^T P^{TT} = P^T AP = P^{-1}AP$$

이므로 symmetric matrix이다.

그리고 $P^{-1}AP$의 첫 번째 column은 다음과 같이 나타낼 수 있다.

$$P^{-1}APe_1 = P^{-1}Av_1 = P^{-1}\lambda_1 v_1 = \lambda_1 P^{-1}v_1 = \lambda_1 e_1 = \lambda_1 \begin{bmatrix}1\\0\\\cdot\\\cdot\\\cdot\\0\end{bmatrix} = \begin{bmatrix}\lambda_1\\0\\\cdot\\\cdot\\\cdot\\0\end{bmatrix}$$

$P^{-1}AP$이 symmetric matrix이므로 $P^{-1}AP$이 $\begin{bmatrix}\lambda_1 & 0\\0 & B\end{bmatrix}$꼴로 나타난다.

(0은 n-1개의 0을 의미하고 B는 $(n-1)\times(n-1)$짜리 symmetric matrix이다)

B가 $(n-1)\times(n-1)$짜리 symmetric matrix이므로 가정에 의해 orthogonally diagonalizable하고 diagonal mat. D'과 $(n-1)\times(n-1)$짜리 orthogonal mat. Q가 존재하여 $QD'Q^{-1}$꼴로 나타내진다.

$R=\begin{bmatrix}1 & 0\\0 & Q\end{bmatrix}$인 R을 설정하면 $\begin{bmatrix}1 & 0\\0 & Q\end{bmatrix}\begin{bmatrix}1 & 0\\0 & Q^{-1}\end{bmatrix}=\begin{bmatrix}1 & 0\\0 & I\end{bmatrix}$으로 R이 invertible하다.

따라서 $R^{-1}=\begin{bmatrix}1 & 0\\0 & Q^{-1}\end{bmatrix}$이고 $R^{-1}=\begin{bmatrix}1 & 0\\0 & Q^{-1}\end{bmatrix}=\begin{bmatrix}1 & 0\\0 & Q^{T}\end{bmatrix}=\begin{bmatrix}1 & 0\\0 & Q\end{bmatrix}^{T}=R^{T}$이므로 R도 orthogonal matrix 가 된다.

$U=PR$인 U를 설정하면 $U^T=(PR)^T=R^TP^T=R^{-1}P^{-1}=U^{-1}$로 U 또한 orthogonal 임을 알 수 있다. 그러면

$$U^{-1}AU=(R^{-1}P^{-1})A(PR)=R^{-1}(P^{-1}AP)R=R^{-1}\begin{bmatrix}\lambda_1 & 0\\0 & B\end{bmatrix}R$$
$$=\begin{bmatrix}1 & 0\\0 & Q^{-1}\end{bmatrix}\begin{bmatrix}\lambda_1 & 0\\0 & B\end{bmatrix}\begin{bmatrix}1 & 0\\0 & Q\end{bmatrix}=\begin{bmatrix}\lambda_1 & 0\\0 & Q^{-1}B\end{bmatrix}\begin{bmatrix}1 & 0\\0 & Q\end{bmatrix}=\begin{bmatrix}\lambda_1 & 0\\0 & D'\end{bmatrix}$$

으로 $n\times n$ matrix가 orthogonally diagonalizable됨을 알 수 있다.

따라서 수학적 귀납법에 의해 square symmetric matrix는 orthogonally diagonalizable 하다는 것을 증명할 수 있다.

3.2. SVD

A는 $m\times n$ matrix라고 하자.

A^TA는 $n\times n$ matrix이므로 이에 대한 eigenvalue / eigenvector를 찾을 수 있다.

A^TA의 eigenvalue를 $\lambda_1, \lambda_2, \cdots \lambda_n$로 설정하자. (W.L.O.G. $\lambda_1 \geq \lambda_2 \cdots \geq \lambda_n$)

Theorem4와 Gram-Schmidt process를 이용해서 찾은 orthonormal eigenvector는 $v_1, v_2, \cdots v_n$로 설정하자.

LEM3. 모든 A^TA의 eigenvalue는 0보다 크거나 같다.

pf〉 A^TA는 symmetric matrix이므로

$A^TAv = \lambda v \Rightarrow v^TA^TAv = \lambda v^Tv \Rightarrow \| Av \|^2 = \lambda \| v \|^2 \Rightarrow \lambda \geqq 0$

LEM4. $rank(A) = rank(A^TA)$

pf〉 $Ax = 0 \Leftrightarrow A^TAx = 0$

pf〉 ($\Rightarrow$) trivial

($\Leftarrow$) $A^TAx = 0 \Rightarrow x^TA^TAx = 0 \Rightarrow \| Ax \|^2 = 0 \Rightarrow Ax = 0$

$null(A^TA) = null(A)$

$rank(A) + null(A) = n$ (A: $m \times n$ mat.)

$rank(A^TA) + \nu ll(A^TA) = n$ (A^TA: $n \times n$ mat.)

$\therefore rank(A) = rank(A^TA)$

LEM5. k가 0보다 큰 eigenvalue의 개수라고 할 때, $k = rank(A) = r$이다.

pf〉 k의 정의에 따라 $\lambda_k > \lambda_{k+1} = 0$이다.

따라서, $\lambda_1 v_1 \cdots \lambda_k v_k > 0$, $\lambda_{k+1} v_{k+1} \cdots \lambda_n v_n = 0$이다.

$k < r$ 이면, $\{\lambda_1 v_1 \cdots \lambda_k v_k\}$가 $Range(A^TA)$를 span하고 $\dim(Range(A^TA)) = r$이므로

$\{\lambda_1 v_1 \cdots \lambda_k v_k\}$가 span할 수 있는 최대 dimension은 k이다.

그러나 $Range(A^TA)$의 dimension은 r로 k보다 크기 때문에 모순이다.

$k > r$ 이면, dimension보다 orthogonal한 vector의 수가 많아지게 된다.

다른 말로 dimension이 r이면, orthogonal한 최대 vector의 수는 r개 인데

$k > r$ 이므로 모순이다.

따라서, $k = r$ 임을 알 수 있다.

$\lambda_1, \lambda_2, \cdots \lambda_n$ 와 $v_1, v_2, \cdots v_n$은 eigenvalues 와 eigenvectors 이므로

$$
\begin{aligned}
A^TAv_1 &= \lambda_1 v_1 \\
A^TAv_2 &= \lambda_2 v_2 \\
&\vdots \\
A^TAv_r &= \lambda_r v_r \\
&\text{----------} \\
A^TAv_{r+1} &= \lambda_{r+1} v_{r+1} \\
&\vdots \\
A^TAv_n &= \lambda_n v_n
\end{aligned}
$$

와 같이 n개의 식을 만들어 볼 수 있다.

이제 Av_i를 생각해 보자.

앞의 LEM4의 증명 과정에서 $Ax = 0 \Leftrightarrow A^TAx = 0$를 증명하였는데 이에 따라 $Av_{r+1} \cdots Av_n$은 모두 0이 됨을 알 수 있다. 따라서 $v_{r+1} \cdots v_n$은 null(A)에 속한다. $v_1 \cdots v_r$은 모두 $v_{r+1} \cdots v_n$와 orthogonal한 vector들이기 때문에 $v_1 \cdots v_r$은 null(A)의 orthogonal complement 인 row(A)에 속한다.

$$v_1 \cdots v_r \in row(A),\ v_{r+1} \cdots v_n \in null(A)$$

따라서 $Av_1 \cdots Av_r$은 $range(A) = col(A)$에 속한다.

이제 Av_i를 orthonormal하게 만들어 볼 것이다. 그러면 각 Av_i를 각각의 norm으로 나누어 주어야 한다. 우리는 Av_i의 norm을 singular value라고 부르고, $\sigma_1, \sigma_2, \cdots \sigma_r$로 표기할 것이다.

$$\sigma_1 = \| Av_1 \|,\ \sigma_2 = \| Av_2 \|,\cdots\ \sigma_r = \| Av_r \|$$

Singular value는 앞서 구했던 A^TA의 eigenvalue와도 연관성이 있다. σ_i^2과 eigenvalue의 특징을 이용하면

$$\sigma_i^2 = \| Av_i \|^2 = v_i^TA^TAv_i = v_i^T\lambda_i v_i = \lambda_i \| v_i \|^2 = \lambda_i \ (\because\ v_i\text{: orthonormal vector})$$
$$\therefore \sigma_i = \sqrt{\lambda_i}$$

로 두 값의 관계를 알 수 있다.

각 Av_i를 norm으로 나누어 만들어진 orthonormal vector들을 $u_1, u_2, \cdots u_r$로 표기하자.

$$u_1 = \frac{Av_1}{\sigma_1},\ u_2 = \frac{Av_2}{\sigma_2},\cdots\ u_r = \frac{Av_r}{\sigma_r}$$

이제 여러 개의 식들을 종합하여 하나의 행렬식으로 만들어 줄 것이다.

$$
\begin{aligned}
Av_1 &= u_1\sigma_1 \\
Av_2 &= u_2\sigma_2 \\
&\vdots \\
Av_r &= u_r\sigma_r \\
&---------- \\
Av_{r+1} &= 0 \\
&\vdots \\
Av_n &= 0
\end{aligned}
$$

위의 식에서 r개의 벡터들을 나타낸 식을 행렬식으로 바꾸어 보면

$$
[A][v_1 v_2 \cdots v_r] = [u_1 u_2 \cdots u_r]\begin{bmatrix} \sigma_1 & & 0 \\ & \sigma_2 & \\ & & \ddots \\ 0 & & \sigma_r \end{bmatrix} \Rightarrow AV = U\Sigma
$$

와 같이 나오게 된다. 여기서 우리는 A를 다른 행렬들의 곱으로 나타내고자 하므로 V를 어떠한 형태를 취하여 우변으로 옮겨주어야 한다. 그러기 위해서는 V가 $n \times n$ matrix 형태가 되어야 한다. 따라서 $null(A)$에 속해있는 $v_{r+1} \cdots v_n$를 추가해 확장시켜 주자. 마찬가지로 U는 $u_1, u_2, \cdots u_r$과 orthogonal한 vector들을 추가시키고 Σ에는 0을 추가시켜 확장시켜 주자. 확장시킨 값들은 곱하였을 때 모두 0을 결과 값으로 가져서 식에 영향을 주지 않는다.

$$
AV = U\Sigma \Rightarrow A\tilde{V} = \tilde{U}\tilde{\Sigma}
$$

확장시킨 행렬은 ~를 이용해서 표기하자. $\tilde{V}$는 orthonormal vector들로 이루어진 행렬이기 때문에 $\tilde{V}\tilde{V}^T = I$임을 알 수 있다. 이를 이용해서 A에 대한 식으로 고쳐보면

$$
A\tilde{V} = \tilde{U}\tilde{\Sigma} \Rightarrow A\tilde{V}\tilde{V}^T = \tilde{U}\tilde{\Sigma}\tilde{V}^T \Rightarrow A = \tilde{U}\tilde{\Sigma}\tilde{V}^T
$$

가 된다. 이 형태를 **full SVD** 라고 부른다.

만약 확장되기 전의 상태처럼 r개의 벡터를 제외한 나머지 값들을 지워주어 처음과 같은 상태를 만들어주게 되면,

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

이 되고 이를 **reduced SVD** 라고 한다.

3.3. SVD의 적용

$A=\begin{bmatrix}0&-1\\1&0\\2&1\end{bmatrix}$ 인 matrix A에 대해 SVD를 적용해 보자.

우선 $A^TA=\begin{bmatrix}0&1&2\\-1&0&1\end{bmatrix}\begin{bmatrix}0&-1\\1&0\\2&1\end{bmatrix}=\begin{bmatrix}5&2\\2&2\end{bmatrix}$ 가 되어

$$\left|\begin{bmatrix}5-\lambda&2\\2&2-\lambda\end{bmatrix}\right|=0 \;\Rightarrow\; \lambda^2-7\lambda+6=0 \;\Leftrightarrow\; \lambda=1\text{ or }6$$

로 eigenvalue를 구할 수 있다.

$$A^TAv_1=\lambda_1v_1,\ A^TAv_2=\lambda_2v_2$$

위의 관계식을 이용하여 eigenvector를 구해준 뒤 각각의 크기로 나눠주면 unit eigenvector인 $v_1=\frac{1}{\sqrt{5}}\begin{bmatrix}-1\\2\end{bmatrix}$, $v_2=\frac{1}{\sqrt{5}}\begin{bmatrix}2\\1\end{bmatrix}$가 된다.

또한 $\sigma_1=\sqrt{\lambda_1}=1\sigma_2=\sqrt{\lambda_2}=\sqrt{6}$ 이 되고, $\Sigma=\begin{bmatrix}1&0\\0&\sqrt{6}\end{bmatrix}$ 이 된다.

$AV=U\Sigma$에서 U를 구해보면

$AV=\frac{1}{\sqrt{5}}\begin{bmatrix}0&-1\\1&0\\2&1\end{bmatrix}\begin{bmatrix}-1&2\\2&1\end{bmatrix}=\frac{1}{\sqrt{5}}\begin{bmatrix}-2&-1\\-1&2\\0&5\end{bmatrix}=U\begin{bmatrix}1&0\\0&\sqrt{6}\end{bmatrix}$ 이고

따라서 $U=\begin{bmatrix}-\frac{2}{\sqrt{5}}&-\frac{1}{\sqrt{30}}\\-\frac{1}{\sqrt{5}}&\frac{2}{\sqrt{30}}\\0&\frac{5}{\sqrt{30}}\end{bmatrix}$ 이다.

마침내 $A=\begin{bmatrix}-\frac{2}{\sqrt{5}}&-\frac{1}{\sqrt{30}}\\-\frac{1}{\sqrt{5}}&\frac{2}{\sqrt{30}}\\0&\frac{5}{\sqrt{30}}\end{bmatrix}\begin{bmatrix}1&0\\0&\sqrt{6}\end{bmatrix}\begin{bmatrix}-\frac{1}{\sqrt{5}}&\frac{2}{\sqrt{5}}\\\frac{2}{\sqrt{5}}&\frac{1}{\sqrt{5}}\end{bmatrix}$ 꼴로 나타낼 수 있다.

4. Least Square Method

주어진 식들의 연립을 통해서 해를 구하려고 할 때, 이를 행렬 곱의 형태로 나타낼 수 있다. 이 때, 식의 개수가 많아 행렬의 행이 열보다 많은 경우, $Ax=b$의 해는 대체로 없다. 그러나 최대한 해에 가까운 값을 찾기 위해서 Least Square Method를 사용한다.

문제 설정) 주어진 $A^{m,n}$ 과 $b \in R^m$ $(m \geqq n \geqq 1)$ 에 대해서 $\| Ax-b \|_2$를 최소화시키는 $x \in R^n$를 구하여라. 이 때, 값을 최소화시키는 벡터 $\vec{x}$는 $Ax=b$의 least square solution라 부른다.

해결)

$\| Ax-b \|_2$ 가 최소가 되는 Ax 는 b를 range(A) 에 projection 시켰을 때이다.

$$\text{Col(A)} = \{Ax \mid x \in R^n\}$$
$$\rightarrow (b-Ax) \perp \text{col(A)} \Leftrightarrow (b-Ax) \perp a_i \ (a_i\text{: A의 i번째 열})$$
$$A^T(b-Ax) = \vec{0} \Rightarrow A^Tb = A^TAx$$

따라서 우리는 $A^TAx = A^Tb$ (Normal equation)을 얻을 수 있다.

A^TA는 n×n 정사각행렬이기 때문에 역행렬을 구할 수 있다.

이것으로 $|A^TA|$가 0이 아닌 경우 normal equation을 해결할 수 있다.

그러나 대부분의 경우에는 역행렬을 구하기 어렵다.

그러므로 행렬의 연산을 돕기 위해 singular value decomposition을 이용해보자.

앞의 SVD를 인용하자면 어떤 행렬 A에 대해 아래와 같은 행렬의 곱으로 나타낼 수 있다는 것이다.

$$A = U\Sigma V^T \ \ U, V\text{: orthogonal} \ \ \Sigma\text{: diagonal}$$

$A^+ = V\Sigma^+ U^T$ 라 하자. $(\Sigma^+ = (\Sigma_r^+ \ 0) \in R^{n \times m}, \Sigma^+ = diag(1/\sigma_1, ...1/\sigma_r, 0, ..., 0) \in R^{n \times n})$
$Ax=b$는 해가 없기 때문에 $R(A)$에 대하여 b를 project시키는 방법을 생각해 보자. 그러면,

$$Ax = AA^{+}b \Leftrightarrow A(x - A^{+}b) = 0$$
$$x = A^{+}b = V\Sigma^{+}U^{T}b$$

이렇게 least square solution을 구하는 방법을 찾을 수 있다.

Ⅲ. 추천 시스템 문제의 설정

기존의 시스템에서 데이터는 어떤 item에 대한 user의 평가로 제시된다. User의 평가는 항상 양의 값이고 어떤 item에 대한 user의 데이터가 존재하지 않는다면 점수는 0점이 된다. Item에 대한 user의 평가를 행렬 R을 통해 나타낼 때, R_{ij}에는 i번째 user가 j번째 item에 부여한 평점이 입력된다. $\overline{R}$를 기존의 데이터로부터 빈 데이터를 복구한 행렬이라 할 때 기존 추천 시스템 문제는 $\| \overline{R} - R \|_F^2$ 을 최소화 하는 $\overline{R}$을 구하는 문제와 같다. 이 점을 볼 때 기존의 시스템은 빈 데이터를 복구하는 데 중점을 두었다고 할 수 있다.

이번에 우리가 새로 고안한 시스템은 개개인의 특성을 예측하는 데 중점을 둔 시스템이다. Item 마다 평가에 중요시되는 요소들 n개를 설정한다. 이 때, user가 제시한 평가와 item에 대한 자료조사를 통해 얻을 수 있는 데이터를 바탕으로 각각의 요소를 수치화시켜 벡터의 형태로 나타낸다. 따라서 시스템 문제는 user의 평가를 n개의 벡터의 합으로 나타낼 때, 개개인의 특성을 나타내는 계수, 즉 각 벡터에 곱해지는 계수를 구하는 문제와 같다. 계수들은 개개인의 특성을 가장 잘 나타낼 수 있는 요소이기 때문에 개인에 최적화된 예상값을 나타낼 수 있다.

Item들에는 여러 가지 종류가 있지만 본 연구에서는 많은 사람들이 자주, 그리고 즐겨 보는 영화를 추천하는 시스템을 만들기로 했다.

Ⅳ. 추천 문제 해결을 위한 모델

1. 요인의 선택

영화의 경우 문화적, 사회적으로 영향을 많이 받는 매체이기 때문에 많은 요인들에 의해 영향을 받게 된다. 그 중에서도 가장 큰 영향을 미치는 요인인 장르, 감독, 배우, 영화의 인지도, 상영 시간으로 선택하였다.

장르의 경우 매우 중요한 요소로 장르에 따라 평가가 크게 바뀌는 것은 당연한 일이라고 할 수 있다. 아무리 영화에 관심이 없는 사람들도 이에 대해서는 확고한 취향을 가지고 있기 때문이다. 감독과 배우의 경우는 감독 특유의 영화 스타일, 배우의 연기 스타일로 인해 영화의 작품성에 영향을 줄 수 있다. 또한 영화의 평가는 인지도에 따라 달라지는 경우가 많다. 상당수의 사람들은 대세를 따라가려는 경향이 있기 때문에 유명하다는 영화를 관람하고 싶어 하고 더 높게 평가해주는 경우가 많다. 반면에 '언더독'효과라고 하여 사람들이 잘 알지는 못하지만 마니아층이 볼만한 유명하지 않은 영화들에 후한 평가를 해 주는 사람도 있다. 따라서 인지도는 개개인에 따라서 평가 기준이 완전히 달라지는 요인이다. 상영 시간의 경우, 시간이 길어질수록 영화의 내용이 방대해지고 스케일이 커질 수 있지만 반대로 이를 루즈하다고 생각하여 집중하기 어려워진다고 생각하는 사람들도 있을 수 있다. 이처럼 상영 시간도 하나의 평가 기준이라고 생각되어 선택하였다.

2. 수량화를 위한 방법

2.1. Baseline Predictor Method

대부분의 수량화를 위해서는 기존에 존재하던 방법을 이용할 것이다. 그 중 하나인 baseline pridictor method를 사용하면 전체 rating, user의 rating, item의 rating을 모두 고려해서 더 정확한 값을 예측할 수 있다. 이 방법은 행렬의 비어있는 값에 대한 추정을 하는 방법이다.

먼저 user가 각 item에 대해 매긴 평가 값을 행렬의 형태로 만들어 주자. i번째 행, j번째 열에 i번째 user가 j번째 item에 매긴 평가를 넣어 주어 행렬 R을 만들자. 이 때 평가하지 않은 값, 즉 비어있는 값을 b_{ij}라고 하자. (i번째 행, j번째 열에 위치한 값)

$$b_{ij} = \mu + b_i + b_j$$

(μ = 전체 data의 rating average

b_i = i 번째 user의 rating average - μ

b_j = j 번째 item의 rating average - μ)

위의 식처럼 b_{ij}를 추정해 주게 되면 전체 rating의 흐름과 user, 그리고 item의 경향을 모두 반영할 수 있어 적합한 결과를 얻을 수 있다. 어느 하나의 요인에 치중된 값이 아니라 여러 가지 요인을 동시에 고려해 준 값이기 때문에 편차도 크지 않을 것으로 예상할 수 있다.

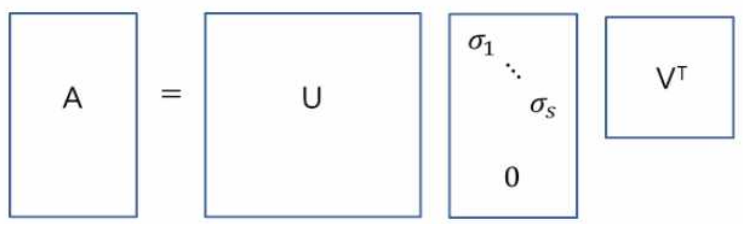
이렇게 구해준 b_{ij}값을 기존의 행렬 R에 대입한 행렬을 R_1이라고 하자.

$$R_1 = \begin{bmatrix} a_{11}\, a_{12} \cdots a_{1n} \\ a_{21}\, b_{22} \cdots a_{2n} \\ \vdots \\ b_{n1}\, a_{n2} \cdots b_{nn} \end{bmatrix}$$

2.2. Matrix Factorization

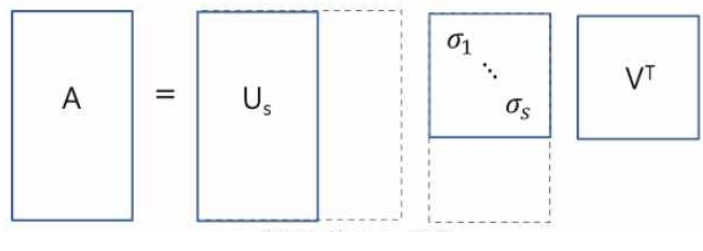
Matrix Factorization을 이용하면 baseline predictor method를 통해 구한 R_1의 노이즈를 제거하고 차원을 줄여 계산을 간결하게 할 수 있다. 우리는 그 중에서 SVD[3,4]를 통해 행렬의 차원을 줄여주는 과정을 거치려고 한다.

아래와 같이 m×n 행렬 A를 SVD로 분해하는 것을 full SVD라 부른다. (단, m 〉 n)

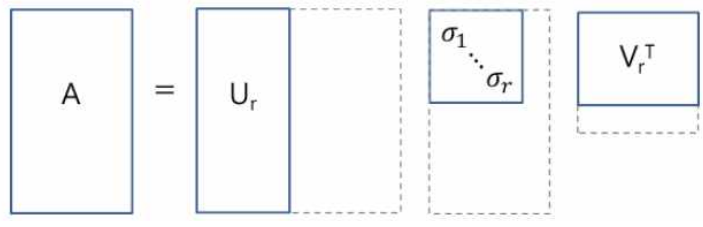


[Fig. 1] Full SVD

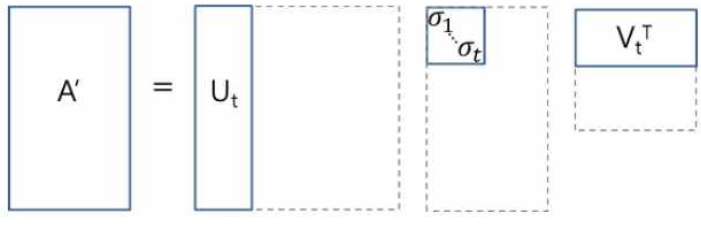
그런데 실제로 full SVD를 사용하하는 경우는 드물며 아래 그림들과 같이 reduced SVD를 사용하는 것이 일반적이다. (s = n 개의 singular values 중 0이 아닌 것들의 개수가 r개, t 〈 r 이라고 가정)



[Fig. 2] Thin SVD



[Fig. 3] Compact SVD



[Fig. 4] Truncated SVD

[Fig. 2] 는 Σ에서 대각선 위가 아닌 0으로 구성된 부분을 없애고 U에서는 아예 대응되는 열 벡터들을 제거했다. [Fig. 3]은 비대각 원소들뿐만 아니라 0인 singular value들까지 모두 제거한 형태이다. (compact SVD) 이 때, 이렇게 계산된 A가 원래의 A와 동일한 행렬이 나옴은 쉽게 확인할 수 있다. 그러나 [Fig. 4]의 경우는 0이 아닌 singular values까지 제거한 형태로서 (truncated SVD) 이 경우에는 원래 A가 보존되지 않고 A에 대한 근사행렬 A′이 나온다. 여기서 특이값의 경우 크기가 큰 순서부터 채택한다.

이렇게 truncated SVD로 근사한 행렬 A′은 matrix norm 을 최소화시키는 rank가 t인 행렬으로 데이터 압축, 노이즈 제거 등에 활용될 수 있다. 이를 R_1에 적용시켜 R_2를 얻어낼 수 있다.

2.3. 유사도 측정

Baseline predictor method, matrix factorization을 데이터의 예측값을 모두 채운 행렬 R_2를 얻어냈다. 행렬 R_2의 평가 값들을 이용하면 동일한 user가 평가한 임의의 두 item의 데이터를 알 수 있는데 이를 비교하면 두 item 간의 유사도를 얻어낼 수 있다.

유사도를 구하는 방법은 크게 두 가지로 나눌 수 있다. 첫 번째는 유사도를 거리의 차로 나타내는 것이다. 두 번째는 유사도를 계수로 나타내는 것이다. 거리로 나타내는 경우에는 거리가 짧을수록 유사도가 높아지고 계수로 나타내는 경우에는 계수가 클수록 유사도가 높아지는 것을 볼 수 있다. 따라서 거리로 나타낼 때에는 그 값의 역수를 취하는 등 연산을 취해주어야 한다.

밑은 둘의 대표적인 예시를 나타낸 것이다.

가) 거리로 나타낸 유사도

1) Euclidian distance $d(x,y)=(\sum_{i=1}^{n}(q_i-p_i)^2)^{\frac{1}{2}}$

2) Minkowski distance $d(x,y)=(\sum_{i=1}^{n}(q_i-p_i)^m)^{\frac{1}{m}}$

3) Mahalanobis distance $d(p,q)=((p-q)^{\Sigma^{-1}}(p-q)^T)^{\frac{1}{2}}$

$$\text{where } \Sigma=\begin{pmatrix} Cov(X_1,X_1) & \cdots & Cov(X_1,X_n) \\ \vdots & \ddots & \\ Cov(X_n,X_1) & \cdots & Cov(X_n,X_n) \end{pmatrix}$$

나) 계수로 나타낸 유사도

1) Cosine similarity $\cos(\theta)=\frac{A \cdot B}{\|A\| \|B\|}$

2) 확장자 카드 계수 $EJ=\frac{x \cdot y}{\|x\|^2+\|y\|^2+x \cdot y}$

이와 같은 유사도 계산법을 이용하여 item 간의 유사도를 구해주면 예상 평점을 구해낼 수 있다. 이번 경우에는 Euclidian distance를 이용하여 유사도를 계산해줄 것이다. 거리로 나타낸 유사도는 그 크기가 클수록 유사도가 줄어들기 때문에 다음과 같이 새로 정의해 주자.

$$s(x,y)=\frac{1}{(\sum_{i=1}^{p}\frac{(x_i-y_i)^2}{p})^{\frac{1}{2}}}$$

Ex) x=(1,2,3) y=(1,3,4) 일 때 s(x,y)는 $\frac{1}{(\frac{0^2+1^2+1^2}{3})^{0.5}}=\frac{\sqrt{3}}{\sqrt{2}}$ 이 된다.

이처럼 유사도를 함수 s를 통해서 구해주고 R_i를 행렬 R_2의 i번째 행이라 하면 item i와 j의 유사도를 밑과 같이 행렬 S를 통해서 나타내자.

$$S_{ij} = \begin{cases} s(R_i, R_j) & \text{if } i \neq j \\ 0 & \text{if } i = j \end{cases}$$

2.4. Collaborative Filtering

2.4.1. User-based collaborative filtering

실시간으로 유사도를 계산하기 때문에 데이터의 양이 작고 데이터 변경이 자주 일어나는 경우에 사용하게 된다. 각 사용자마다 그들의 행렬값을 가지고 있는 경우가 많다.

2.4.2 Item-based collaborative filtering

항목 간 유사도를 저장한 뒤에 가중평균을 내는 방식이므로 데이터의 크기가 큰 경우에도 사용할 수 있다.

수량화를 하고자 하는 목적은 item들 간의 관계를 통해 각 요인들에 주어지는 값을 정해주기 위함이므로 item-based collaborative filtering에 초점을 맞추기로 결정하였다.

앞선 과정을 통해서 item i와 j의 유사도를 얻어낼 수 있었다. 따라서 유사도를 통해 임의의 user i에 대한 item j의 평가값을 추정할 수 있다. 따라서 행렬 R의 빈 데이터는 유사도를 가중치로 둔 user의 데이터의 가중 평균으로 다시 한 번 생각해볼 수 있다. 따라서

$$\overline{R_{ij}} = \frac{\sum_{k=1}^{n} S_{kj} R_{ik}}{\sum_{k=1}^{n} S_{kj}}$$

로 표현된다.

3. 수량화를 통한 벡터 설정

수량화한 값을 행렬화시켜서 개인에 최적화된 추천 시스템을 위한 5 개의 벡터를 만들고자 한다. 이들을 감독, 배우, 장르, 인지도, 상영시간 순으로 e_1, e_2, e_3, e_4, e_5 라고 하자. 이 벡터들은 user i의 각 영화에 대한 각 요인의 수량화된 값을 나타낸 벡터들로 n개의 영화에 대한 값들이라면 n×1 행렬이다. 즉, e_i의 j번째 행에는 j번째 영화의 i번째 요인의 수량화된 값이 들어가게 된다.

3.1. 감독, 배우, 장르에 따른 벡터 설정

감독, 배우, 장르게 대해서는 앞의 유사도를 이용한 방법을 사용할 것이다.

먼저, 감독의 경우 감독 A가 찍은 영화의 평가에 사용자가 참여한 경우와 그렇지 않은 경우로 나눌 수 있다. 참여했다면 참여한 n개의 영화에 매긴 평점을 $a_1,a_2,a_3...a_n$라 하자. 이 때, 감독 A가 만든 다른 영화들 중 평가하지 않은 경우에는 그 감독에게 평점들의 평균인

$$d_i = \frac{a_1 + a_2 + ...a_n}{n}$$

와 같은 값을 부여하자.

그러나 사용자가 감독 A가 만든 영화를 한 번도 보지 않았을 경우도 충분히 존재한다. 따라서 이 경우에는 유사도를 이용한 수량화 방법을 사용할 것이다. 데이터 값들 중에서 각각의 사용자에 대해서 감독들을 평가한 데이터들(앞에서 산술 평균으로 나타낸 값)과 아직 그 감독이 만든 영화를 보지 않아서 비어있는 데이터값을 가지는 행렬을 얻을 수 있다. 주어진 데이터들을 바탕으로 baseline predictor method를 적용하면 행렬의 빈 자리를 채워 각 항이 $b_{11},b_{12},...b_{ij}$인 행렬이 되고 matrix factorization을 적용하여 $c_{11},c_{12},...c_{ij}$로 나타낼 수 있다. 이후, 이 데이터들을 통해 각 감독 간의 평가 유사도를 구해주게 된다. 그러면 i번째 감독과 j번째 감독의 유사도는 s_{ij}로 나타낼 수 있고 이를 바탕으로 k번째 감독의 데이터는

$$u_k = \frac{s_{k1}c_{k1} + s_{k2}c_{k2} + ... + s_{kn}c_{kn}}{s_{k1} + s_{k2} + ...s_{kn}}$$

으로 가중평균을 내주어 구해 줄 수 있다.

감독 i에 대한 수량화 값을 f_i라고 할 때, 사용자가 i가 만든 영화에 평점을 남겼을 경우 $f_i = d_i$로 원래의 값을 부여해 주고 그렇지 않은 경우에는 $f_k = u_k$로 가중평균을 내준 값을 부여해 주자. 이 값들을 각 영화에 대해서 적합한 값을 부여해 주게 되면 e_1을 구할 수 있다.

배우, 장르에 대해서도 같은 방법으로 벡터들을 구할 수 있다. 만약 감독이 여러 명이거나, 배우가 한 명이 아니거나, 장르가 두 가지 이상에 걸쳐 있는 등 한 가지 요인에서 두 가지 이상의 값을 사용해야 한다면 그들의 산술 평균을 내어 사용하자. 예를 들어 어떤 영화 i에 출연한 배우들의 데이터를 $j_1,j_2,..j_n$라 하면, e_2에 들어가게 될 데이터는

$$e_{i2} = \frac{j_1 + j_2 + \cdots j_n}{n}$$

와 같은 산술 평균을 낸 값이 된다.

3.2. 인지도에 따른 벡터 설정

영화의 인지도를 나타낼 수 있는 요인들에는 관객 수, 예고편 조회수 등이 있다. 그러나 오래된 영화 중에서 예고편이 없었던 영화가 많았기 때문에 적합하지 않다고 생각했다. 따라서 실시간으로 확실하게 주어지는 데이터인 관객 수를 바탕으로 인지도를 수량화하기로 결정했다.

이 때, 중요하게 생각해 보아야 할 요소는 년도에 따라서 관객 수가 차이날 수 있다는 것이다. 영화는 사회적인 영향을 많이 받는 매체이기 때문에 경기 사정이 좋지 않다면 관객 수가 줄어들 수도 있고 현재 영화 산업이 더 발전했기 때문에 1950년대 영화와는 엄청난 관객 수 차이를 보일 수밖에 없다. 따라서 이 문제를 해결하기 위해서 영화 i의 관객 수를 p_i, 그 영화가 개봉한 해인 j의 영화당 평균 관객 수를 m_j라고 할 때,

$$e_{i4} = \frac{p_i}{m_j}$$

를 채택하기로 하였다.

이 외에도 영화의 상영이 끝나지 않았을 경우를 생각해 주어야 한다. 만약 개봉한지 얼마 되지 않은 영화의 경우에는 관객 수가 작을 수밖에 없다. 이 문제를 해결하기 위해서 개봉 일차에 따른 영화 관객 수의 통상적인 증가 추이를 통해 개봉 일차에 따라 상영이 종료되었을 때 관객 수를 예측하고자 하였다.

영화진흥위원회에서의 통계 자료[7]를 바탕으로 총 네 개의 영화에 대한 개봉일차별 누적 관객 수를 그래프로 나타내었다. 그 후에 2차 곡선으로 추세선을 그린 결과는 과 같다.

대부분의 영화의 경우 개봉 0일차에 대해서 누적 관객 수를 0명이라고 할 때,

$$y = -0.0009x^2 + ax$$

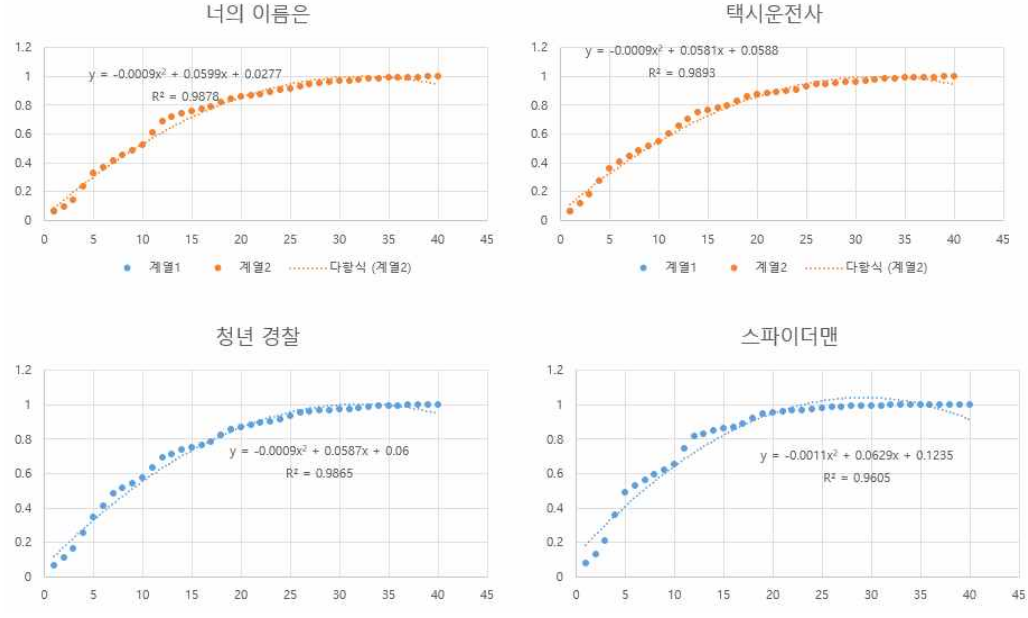
의 형태로 나타난다는 것을 알 수 있다. 따라서 만약 영화 i의 개봉 x일차 누적 관객 수를 알 수 있다면 위의 식에 대입하여 상수 a의 값을 알아낼 수 있고, 개봉 40일차 이후에는 누적 관객 수가 거의 늘어나지 않는 것으로 보아

$$\lim_{x \to 40}(-0.0009x^2 + ax)$$

로 예상 관객 수를 추정할 수 있다. 이 때 예상 관객 수를 q_i라고 한다면

$$e_{i4} = \frac{q_i}{m_j}$$

와 같은 값이 나온다는 것을 알 수 있다.



[Fig. 5] 개봉 일차별 누적 관객 수

3.3. 상영시간에 따른 벡터 설정

상영시간의 경우에는 사용자마다 생각하는 것이 다르다. 어떤 이는 긴 영화를 선호할 수도 있고 어떤 이들은 짧고 강렬하게 끝나는 영화를 선호할 수도 있다. 따라서 상영시간을 그대로 반영하기로 하였다. 기준은 한 시간을 1로 두어 만약 영화가 총 x분이라면

$$e_{j5} = \frac{x}{60}$$

의 형태로 값이 주어지게 된다.

4. 추천 시스템

e_1, e_2, e_3, e_4, e_5를 column으로 하는 n×5 행렬 E를 생각하자.

$$E = (e_1, e_2, e_3, e_4, e_5)$$

주어진 평점의 데이터를 n×1 행렬 $\vec{b}$라 하면 5개의 벡터의 계수를 찾는 문제는

$$E\vec{x} = \vec{b} \text{ where } \vec{x} = \begin{pmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \end{pmatrix}$$

의 해를 찾는 문제와 같다. 행렬 E는 행이 열보다 많은 행렬이므로 적절한 해를 찾기 위해서 Least Square Method를 사용하자.

먼저 SVD를 사용하면

$$E = U\Sigma V^T \ (U, V : \text{orthogonal} \ \ \Sigma : \text{diagonal})$$

로 나타낼 수 있다. 이 때, E의 pseudo inverse를 $E^+ = V\Sigma^+ U^T$와 같이 정의하면

$$\vec{x} = E^+\vec{b} = V\Sigma^+ U^T\vec{b}$$

로 5개의 벡터의 계수를 구할 수 있다.

이제 적당한 계수 c_1, c_2, c_3, c_4, c_5를 구했으므로 원하는 영화에 대한 사용자의 평점을 계산해 낼 수 있다. 어떤 영화 A에 대해서 e_1, e_2, e_3, e_4, e_5에서 A에 해당하는 값을 p_1, p_2, p_3, p_4, p_5라 하면 예상되는 평점은

$$(\text{예상 평점}) = c_1p_1 + c_2p_2 + c_3p_3 + c_4p_4 + c_5p_5$$

로 구해낼 수 있다. 따라서 사용자의 특성에 최적화된 평점을 제시할 수 있는 시스템이라고 할 수 있다.

V. 결론 및 향후 계획

우리는 사용자가 영화를 평가하는 기준에 크게 다섯 가지 요인, 영화의 감독, 배우, 장르, 상영시간, 인지도가 있다고 생각하여 평점 또한 이 5가지 정보의 합으로 보았다. 사용자마다 각 요인의 중요도를 판별하기 위해 이미 평가된 별점과 5가지 정보에 대해 수치화한 데이터 사이의 행렬 방정식을 풀어 가중치를 구했다. 앞서 구한 가중치와 5가지 정보를 이용해 우리는 평가하지 않은 영화에 대한 예상 별점을 얻을 수 있게 된다. 이리하여 우리는 사용자 개개인의 영화에 대한 취향과 별점을 주는 기준을 모두 고려한 추천 시스템을 만들 수 있었다.

실제로 우리가 구한 방법 이외에 다양한 방법을 통해 데이터를 구하고 유사도를 구할 수 있는데 gradient descent method, cosine similarity 등을 이용한 알고리즘과 효율성을 비교해 볼 것이다.

더 나아가 이 알고리즘은 영화 추천뿐만 아니라 개인의 취향이 갈리는 모든 선택에서 적용시킬 수 있는 추천 시스템으로 발전시킬 수 있다. 응용될 수 있는 대표적인 예시로 휴대폰 어플리케이션, 맛집 추천 등이 있으며 이에 도전해볼 계획이다.

VI. 참고문헌

[1] Koren, Y & Bell, R & Volinsky, C. *Matrix Factorization Techniques for Recommender Systems*. Computer, vol. 42, issue 8, 42-49, (2009)

[2] Boyd, S & Vandenberghe, L. *Convex Optimization*, Cambridge University Press. (2004)

[3] Anton, H & Rorress, C. *Elementary Linear Algebra with Supplemental Applications.* Hoboken: Wiley. (2011)

[4] Strang, G. *Linear Algebra and its Application.* Forth Worth: Hartcourt Brace Jovanvich Inc. (1988)

[5] Trefethen, L & Bau, D. *Numerical Linear Algebra*, SIAM. (1997)

[6] Ekstrand, D & Riedl, J & Konstan, J. *Collaborative Filtering Recommender Systems,* Foundation and Trends in Human-Computer Interaction, vol. 4, no. 2, 81-173, (2010)

[7] 영화진흥위원회(KOFIC), http://www.kobis.or.kr/kobis/business/main/main.do

ABSTRACT

Research on Efficient Recommender System Using Matrix of Data

Reseacher : Suhyun Ryu(sophomore, danny906@naver.com)
Jiho Park(sophomore, hy7720@naver.com)
Sehwa Jeong(sophomore, newton0512@naver.com)

Supervisor : Yeongcheol Jeon(Korea Science Academy of KAIST, ycjeon@kaist.ac.kr)

Abstract

In this study, we designed a recommender system that emphasizes individual characteristics. Based on the evaluations presented by the user, it is possible to matrixize the values and make vectors, $e_1, e_2, \cdots, e_n$, corresponding to the most important n factors of the item. Then, the coefficients given to the respective vectors can be predicted using Least Square Method and Singular Value Decomposition (SVD). These coefficients are given differently for each individual and can be said to be an optimized value.

Key Words : Recommender system, Matrixize of data, Least Square Method, SVD