April 21 2015

LECTURE # 10

$$N = \{\perp \cup R\}^{m \times n} \qquad k \ll n \ll m$$

Netflix Matrix

m = # users
n = # movies
k = # (implict features)

Insight: Rows of an $m \times n$ matrix
$\equiv$ m points in an n-dimensial space
Explicit features of a user
$\equiv$ His utilities/rating of each move.
$\Rightarrow$ Dimension Reduction
to just k features.

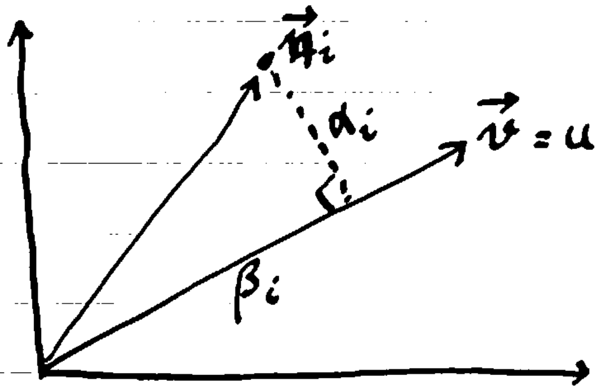
- Find the "best" k-dimensional subspace with respect to the set of points
  ↳ As represent by $U \in \mathbb{R}^{m \times k}$

"BEST" $\equiv$ Minimize the SOS (Sum-of-Squares) of the perpendicular distances of the points to the subspace.

Start with $k = 1$.

1-dim subspace = A line through origin.

BEST LEAST SQUARE FIT.



minimize $\sum_i \alpha_i^2$

$\equiv$ maximize $\sum_i \beta_i^2$

$(\because \vec{n}_i = \text{const.})$

$\vec{n}_i = i^{th}$ row of $N \in \mathbb{R}^n$

$\beta_i = |\vec{n}_i \cdot \vec{v}|$

$\therefore \sum_i \beta_i^2 = |N\vec{v}|^2$ = Sum of length squared of the projections.

First Singular Vector $\vec{V}_1$ of N

$$\vec{V}_1 = \underset{\substack{|\vec{v}|=1 \\ |v|=1}}{\arg\max} \; |N\vec{v}|$$

First Singular Value σ_1 of N

$$\sigma_1 = |N v_1|$$

$$\Rightarrow \sigma_1^2 = \sum_{i=1}^{m} (n_i \cdot v_1)^2 = \sum_i \beta_i^2$$

$\left\{ \right.$ SOS of the projections of the points to line determined by v_1

A Greedy Approach:

$$\vec{v}_2 = \underset{\substack{\vec{v} \perp v_1 \\ |\vec{v}|=1}}{\arg\max} |Nv|; \qquad \sigma_2 = |N\vec{v}_2|$$

Second Singular Vector — Second Singular Value.

The 2-dim subspace spanned by the unit vectors $\vec{v}_1$ and $\vec{v}_2$
⇒ Maximizes SOS of the projections of the m points w.r.t. span$(\vec{v}_1, \vec{v}_2)$

⋮

Continuing ...

$$\vec{v}_p = \underset{\substack{\vec{v} \perp \vec{v}_1, \dots, \vec{v}_{p-1} \\ |\vec{v}|=1}}{\arg\max} |Nv| \qquad \sigma_p = |N\vec{v}_p|$$

p^{th} Singular Vector — p^{th} Singular Value.

r: rank of N. Then the process stops with

$$\vec{v}_1, \vec{v}_2, \dots, \vec{v}_r$$

as the singular vector (orthonormal) and

$$\sigma_1, \sigma_2, \dots, \sigma_r$$

as singular values &

$$\underset{\substack{\vec{v} \perp v_1, \dots v_r \\ |\vec{v}|=1}}{\arg\max} |Nv| = 0.$$

Forbenius Norm of N

$$\|N\|_F = \sqrt{\sum_{jk} n_{jk}^2}$$

<u>Theorem</u>: Let $N \in \mathbb{R}^{m \times n}$ ~~an~~ be an $m \times n$ matrix with singular vectors (resp. singular values)

$$\vec{v}_1, \vec{v}_2, \dots, \vec{v}_r$$

$$(\text{resp. } \sigma_1, \sigma_2, \dots, \sigma_r)$$

⟨1⟩ $V_k = \text{Span}(\vec{v}_1, \vec{v}_2, \dots, \vec{v}_k)$

V_k: Best-Fit k-dim subspace of N.

⟨2⟩ $\|N\|_F = \sum_{i=1}^{r} \sigma_i^2$.

⟨3⟩ $\vec{u}_i = \frac{1}{\sigma_i} N \vec{v}_i$

(Thes are the ~~sig~~ left singular vectors correponding to the right singular vectors $\vec{v}_i$'s).

$U = (\vec{u}_1, \dots, \vec{u}_r)$ $m \times r$

$V = (\vec{v}_1, \dots, \vec{v}_r)$ $n \times r$

$D = \text{Diag}(\sigma_1, \dots, \sigma_r)$ $r \times r$

$$\Rightarrow \quad A = \sum_{i=1}^{r} \sigma_i \vec{u}_i v_i^T = UDV^T$$

<4> $\tilde{U}_k = (u_1, \ldots, u_k)$ $m \times k$ ($k \le r$)

$\tilde{V}_k = (v_1, \ldots, v_k)$ $n \times k$

$\tilde{D}_k = \mathrm{Diag}(\sigma_1, \ldots, \sigma_k)$ $k \times k$

$$\tilde{N} = N_k = \sum_{i=1}^{k} \sigma_i \vec{u}_i \vec{v}_i^T = \tilde{U}_k D_k \tilde{V}_k^T$$

= Truncated Sum.

$$\| N - N_k \|_2^2 = \sigma_{k+1}^2$$

$$\forall_{B,\, \mathrm{rank}(B) = k} \quad \| N - N_k \|_F \le \| N - B \|_F .$$

N_k = "Best" rank-k approximation of A

↳ w.r.t. $\mathcal{L}_2$ or Forbenius norm.

Computing SVD Efficiently.

Repeated Squaring:

$$N = \sum_i \sigma_i u_i v_i^T$$

$$P = N^T N = \sum_{i,j} \sigma_i \sigma_j \left(v_i (u_i^T u_j) v_j^T \right)$$

$$= \sum \sigma_i^2 v_i v_i^T$$

$\vdots$

$$P^2 = \sum \sigma_i^4 v_i v_i^T$$

$$P^k = \sum \sigma_i^{2k} v_i v_i^T = \sigma_1^{2k} v_1 v_1^T + \sum_{i \ge 2} \sigma_i^{2k} v_i v_i^T$$

$$\approx \sigma_1^{2k} v_1 v_1^T .$$

$\therefore \dfrac{P^k}{\|P^k\|_F}$ converges to $v_1 v_1^T$

A rank-1 matrix

Recover $v_1 \rightarrow$ By normalizing the first column to be a unit vector.

$$\left.\begin{array}{l} \sigma_1 = |N v_1| \\ u_1 = \dfrac{1}{\sigma_1} N v_1 \end{array}\right\} \tilde{N} = N_1 = \sigma_1 u_1 v_1^T$$

$\boxed{\text{Repeat}}$ With $N^{(1)} = N - N_1$

Problem N = Netflix matrix is sparse

But $N^T N = P =$ Dense

New Trick: $x_0 \in \mathcal{N}(0, I)$

Random Vector

$$s = \text{Large} = \frac{1}{2\lambda} \log\left(4 \log(2m/\delta)/\epsilon\delta\right)$$

$$\lambda = \min_{i<j} \log\left(\frac{\sigma_i}{\sigma_j}\right)$$

$$x_0 \rightarrow x_1 = N^T(N x_0) \rightarrow x_2 = N^T(N x_1)$$

$$\cdots \longrightarrow x_s = N^T(N x_{s-1})$$

$$\vec{v}_1 = \frac{x_s}{\|x_s\|}; \quad \sigma_1 = |N v_1|; \; u_1 = \frac{N v_1}{\sigma_1}$$

Why: $\quad x_0 = \sum_{i=1}^{r} c_i v_i$

$$x_s = \sigma_1^{2s} v_1 v_1^T \sum_{i=1}^{r} c_i v_i$$

$$\approx c_1 \sigma_1^{2s} v_1$$