Lecture 10

SVD in Least Squares

Principal Components Regression

1. Reduce dimension of each sample from p to $k < p$.

   Let $X \in \mathbb{R}^{n \times p}$ = training features and $U \Sigma V^T = \text{SVD}(X)$. Let V_k = 1^{st} k columns of V.

   Let $\underline{z}_i = V_k^T \underline{x}_i = (i^{th}$ col of $\Sigma_k U_k^T)$

2. Train ML model on $\underline{z}_i$'s

   $$\hat{\underline{b}} = \underset{\underline{b}}{\arg\min} \; \|\underline{y} - Z\underline{b}\|_2^2 = (Z^T Z)^{-1} Z^T \underline{y} \quad \text{where } Z = \begin{bmatrix} - & \underline{z}_1^T & - \\ - & \underline{z}_2^T & - \\ & \vdots & \\ - & \underline{z}_n^T & - \end{bmatrix} \in \mathbb{R}^{n \times k}$$

   usually k is selected so k features are linearly independent (i.e all k σ_i's in Σ_k are > 0)

3. Predict for new sample $\underline{x}_{new}$.

   $\underline{z}_{new} = V_k^T \underline{x}_{new}$

   $\hat{y}_{new} = \underline{z}_{new}^T \hat{\underline{b}}$

4. (Optional) Find equivalent $\underline{w}$ so that $X\underline{w} \approx Z\underline{b}$

   $z_i^T = i^{th}$ row of $U_k \Sigma_k \iff Z = U_k \Sigma_k \Rightarrow Z\underline{b} = U_k \Sigma_k b = U_k \Sigma_k V_k^T V_k b = X_k \; V_k \underline{b}$

   $\Rightarrow Z\hat{\underline{b}} = X_k \hat{\underline{w}}$ where $\hat{\underline{w}} = V_k \hat{\underline{b}}$

   also, $\hat{\underline{b}} = (Z^T Z)^{-1} Z^T y$

   $$= (\Sigma_k U_k^T U_k \Sigma_k^T)^{-1} \Sigma_k U_k^T y$$

   $$= (\Sigma_k \Sigma_k^T)^{-1} \Sigma_k U_k^T y$$

   $$= \Sigma_k^{+} U_k^T y$$

$\Rightarrow \hat{\underline{w}} = V_k \Sigma_k^{+} U_k^T \underline{y}$ $\Leftarrow$ PCR same as projecting each x_i onto best k-dim subspace, then taking pseudoinverse.

Principal Components Crime

Broke

1. Run PCA on all n training samples, mapping $\underbrace{\underline{x}_i}_{\in\mathbb{R}^p} = V_k^T \underbrace{\underline{z}_i}_{\in\mathbb{R}^k}$, $i=1,\ldots,n$

> Specifically V_k is 1st k cols of V matrix from SVD of $X \in \mathbb{R}^{n\times p}$

2. Split data into train set and test set.
3. Train ML model on $(\underline{z}_i, y_i)$, $i=1,\ldots,n_{train}$
4. Measure accuracy on $\underbrace{(\underline{z}_i, y_i),\ i = n_{train+1},\ldots,n}_{\text{test set}}$

> Problem: V_k depends on test data.
> Learned model depends on test data.
> Estimated accuracy artificially large

This really happens, especially in large organizations with different teams doing data prep vs training

Woke

1. Split data into train and test sets
2. Run PCA on n_{train} training samples, mapping $\underline{x}_i \rightarrow \underline{z}_i = V_k^T \underline{x}_i$ for $i=1,\ldots,n_{train}$

> Now V_k is 1st k cols of V matrix from SVD of 1st n_{train} rows of X only

3. Train ML model on $(\underline{z}_i, y_i)$, $i=1,\ldots,n_{train}$
4. Measure accuracy on $\underbrace{(\underline{z}_i, y_i),\ i = n_{train+1},\ldots,n}_{\text{test set}}$

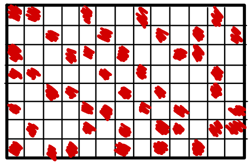
where $\underline{z}_i = V_k^T \underline{x}_i$

→ much more reliable predictor of accuracy

Matrix Completion

$X \in \mathbb{R}^{n \times p}$ (e.g. n movies, p customers) assume X is low rank

only observe subset of entries of X, want to fill in remainder



$\rightarrow X$

$\Omega = \{(i,j) \text{ s.t. } X_{ij} \text{ is observed}\}$

"Ideal":

$$\hat{X} = \underset{M}{\arg\min}\ \text{rank}(M) \quad \text{s.t. } M_{ij} = X_{ij}\ \forall (i,j) \in \Omega$$

- find lowest rank matrix that fits observed entries

$\boxed{\text{intractable}}$

$\text{rank}(X) = \#\{i : \sigma_i > 0\}$

$\|X\|_* = \sum_i \sigma_i$

= trace norm

= nuclear norm

Tractable alternative:

$$\hat{X} = \underset{M}{\arg\min}\ \|M\|_* \quad \text{s.t. } M_{ij} = X_{ij}\ \forall (i,j) \in \Omega$$

or, if data is noisy

$$\hat{X} = \underset{M}{\arg\min}\ \|X_\Omega - M_\Omega\|_2^2 + \lambda \|M\|_*$$

Algorithm:

Iterative Singular Value Thresholding

initialize: $\hat{X} = \text{zeros}(n,p)$

$\hat{X}_\Omega = X_\Omega$ $\leftarrow$ fill in obs. entries

set threshold

for $k = 1, 2, \ldots$

$\hat{X}_{old} = \hat{X}$

$[U, S, V] = \text{svd}(\hat{X})$;

$\hat{S} = S .* (S \geq \text{threshold})$

$\hat{X} = U \cdot \hat{S} \cdot V^T$;

$\hat{X}_\Omega = X_\Omega$;

if $\|\hat{X} - \hat{X}_{old}\|_F < \varepsilon$, stop

end

Eigendecomposition and Page Rank

Given a matrix A, we say a <u>non zero</u> vector v is a eigenvector of A if
↳ $n \times n$

$$\boxed{Av = \lambda v}$$ where λ is a scalar

Let $V = [v_1, v_2, \ldots, v_n]$ be a matrix whose columns are all eigenvectors of A.

$$AV = A[v_1, v_2, \ldots, v_n] = [Av_1, Av_2, \ldots, Av_n] = \underbrace{[\lambda_1 v_1, \lambda_2 v_2, \ldots, \lambda_n v_n]}_{[v_1 \cdots v_n]\begin{bmatrix} \lambda_1 & & & 0 \\ & \lambda_2 & & \\ & & \ddots & \\ 0 & & & \lambda_n \end{bmatrix}}$$

$$\Rightarrow \boxed{AV = V\underline{\Lambda}}$$

If the eigenvectors are <u>Linearly independent</u>

then V is invertible

$$\Rightarrow AVV^{-1} = V\underline{\Lambda}V^{-1} \Rightarrow \boxed{A = V\underline{\Lambda}V^{-1}}$$

Take a matrix X with SVD $X = U\Sigma V^T$

Then $A = \underline{X^TX} = (U\Sigma V^T)^T(U\Sigma V^T) = V\Sigma \underbrace{U^TU}_{I}\Sigma V^T = V\Sigma\Sigma V^T = V\Sigma^2 V^T$

$$\begin{bmatrix}\sigma_1 & & & \\ & \sigma_2 & & \\ & & \ddots & \\ & & & \sigma_n\end{bmatrix}\begin{bmatrix}\sigma_1 & & & \\ & \sigma_2 & & \\ & & \ddots & \\ & & & \sigma_n\end{bmatrix} = \begin{bmatrix}\sigma_1^2 & & & \\ & \sigma_2^2 & & \\ & & \ddots & \\ & & & \sigma_n^2\end{bmatrix}$$

Recall that V is orthogonal $\Rightarrow V^T = V^{-1}$

$\Rightarrow A = X^TX = V\underbrace{\Sigma^2}_{\Lambda}V^{-1}$

$\Rightarrow A = V\Lambda V^T$ = eigendecomposition of A

$\Rightarrow$ eigenvectors of A are the right singular vectors of X

eigenvalues of A are the <u>squared</u> singular values of X

for general A, if A is real and symmetric, then we can ensure the eigenvectors are real and orthonormal $\Rightarrow \underline{V^{-1} = V^T}$

If we can write A as X^TX for some X, then A is real + symmetric

↳ real

Eigenvalues + Eigenvectors

A vector $\underline{v}_k$ (with $\|v_k\|=1$) is an eigenvector of A if there is a scalar λ_k such that
$$A\underline{v}_k = \lambda_k \underline{v}_k$$

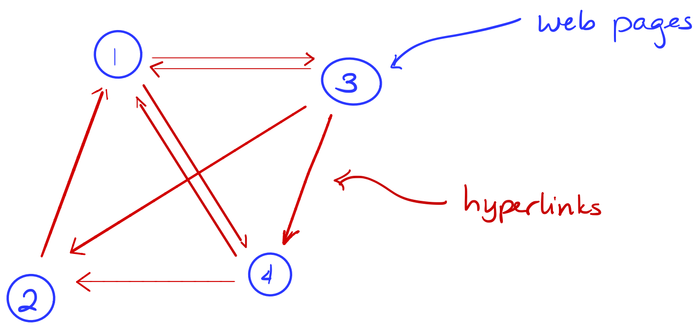
Let $V = [v_1 \; v_2 \; \cdots \; v_n]$ be the n eigenvectors of A. Then

$$AV = A[v_1 \; v_2 \; \cdots \; v_k] = [\lambda_1 \underline{v}_1 \;\; \lambda_2 \underline{v}_2 \; \cdots \; \lambda_n \underline{v}_n]$$
$$= \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{bmatrix} V = \Lambda V$$

$$\boxed{\begin{aligned} &\text{SVD: } X = U\Sigma V^T \in \mathbb{R}^{p \times n} \\ &\text{Let } A = X^T X = V\Sigma U^T U \Sigma V^T \\ &\quad = V \Sigma^2 V^T =: V \Lambda V^T \end{aligned}}$$

$\Longrightarrow$ eigenvalues are real and eigenvectors are orthonormal

PageRank



Imagine surfing web by randomly clicking on links at each page at which you arrive.

You will visit more "important" web pages more frequently.

If you do this long enough, you'll reach a steady state where π_i is the probability you're at page i at any given time.

Let M be adjacency matrix of links and A its column normalized version (Markov matrix)

$$M = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \end{bmatrix} \qquad A = \begin{bmatrix} 0 & 1 & \frac{1}{3} & \frac{1}{2} \\ 0 & 0 & \frac{1}{3} & \frac{1}{2} \\ \frac{1}{2} & 0 & 0 & 0 \\ \frac{1}{2} & 0 & \frac{1}{3} & 0 \end{bmatrix}$$

$\Longleftrightarrow$ $M_{ij} = \begin{cases} 1 & \text{if page } j \text{ links to page } i \\ 0 & \text{otherwise} \end{cases}$

Think about

A_{ij} = Prob (visit site i next given we're at site j now)

We want to find a vector of probabilities $[\pi_1, \pi_2, \pi_3, \pi_4]^T$ so that $\underline{\pi} = A\underline{\pi}$

$\Rightarrow$ $\underline{\pi}$ is simply the leading eigenvector of A !

To see this, let $\pi = v_1$ (1st col. of V). Then

$$\pi = A\pi$$

$$\Rightarrow v_1^T \pi = v_1^T A \pi = v_1^T V \Lambda V^T \pi$$

$$\Rightarrow v_1^T v_1 = 1 = e_1^T \Lambda e_1 = \lambda_1$$

Markov matrices are special: we know $\lambda_1 = 1$ and $\lambda_1 > \lambda_2 \geq \lambda_3 \geq \cdots \geq \lambda_n$

We can find this vector using <u>Power Iterations</u>.

Let $\underline{\pi}^{(0)}$ be an initial guess of π.

for $k = 1, 2, \ldots$

compute $\underline{\pi}^{(k)} = \dfrac{A\pi^{(k-1)}}{\|A\pi^{(k-1)}\|_2}$

using the 2-norm is the denominiator is standard; it ensures the final $\pi^{(k)}$ has norm $\|\pi^{(k)}\|_2 = 1$

for page rank, we might prefer $\|\pi^{(k)}\|_1 = \sum_i \pi_i^{(k)} = 1$; we can manage this 3 ways

1. $\pi^{(k)} = A\pi^{(k-1)} / \|A\pi^{(k-1)}\|_1$
2. use 2-norm in denominator; at end, take $\dfrac{\pi^{(k)}}{\|\pi^{(k)}\|_1}$ as final est.
3. start with $\|\pi^{(0)}\|_1 = 1$ and A column normalized; then don't need denominator at all

Why does this work? We show this under the assumption that the eigenvectors are orthonormal (Otherwise proof is more complex)

We can write $\underline{\pi}^{(0)} = c_1 \underline{v}_1 + c_2 \underline{v}_2 + \cdots + c_n \underline{v}_n$ for some $c_1, \ldots, c_n$, with $c_1 \neq 0$, where $\underline{v}_i = i^{th}$ col of V

now

$$\underline{\pi}^{(k)} \propto A\underline{\pi}^{(k-1)} \propto A^2 \underline{\pi}^{(k-2)} \propto \cdots A^k \underline{\pi}^{(0)}$$

$$= (V\Lambda V^T)^k \underline{\pi}^{(0)} = V\Lambda^k V^T \underline{\pi}^{(0)} = V\Lambda^k V^T (c_1 \underline{v}_1 + \cdots + c_n \underline{v}_n) = V\Lambda^k (c_1 \underline{e}_1 + \cdots + c_n \underline{e}_n)$$

$$= c_1 \lambda_1^k \underline{v}_1 + \cdots + c_n \lambda_n^k \underline{v}_n = c_1 \lambda_1^k \left(\underline{v}_1 + \frac{c_2 \lambda_2^k}{c_1 \lambda_1^k} v_2 + \cdots + \frac{c_n \lambda_n^k}{c_1 \lambda_1^k} v_n \right)$$

as $k \to \infty$, $\dfrac{\lambda_i^k}{\lambda_1^k} \to 0$ for $i \neq 1$

so $A^k \underline{\pi}^{(0)} \to c_1 \lambda_1^k \underline{v}_1$

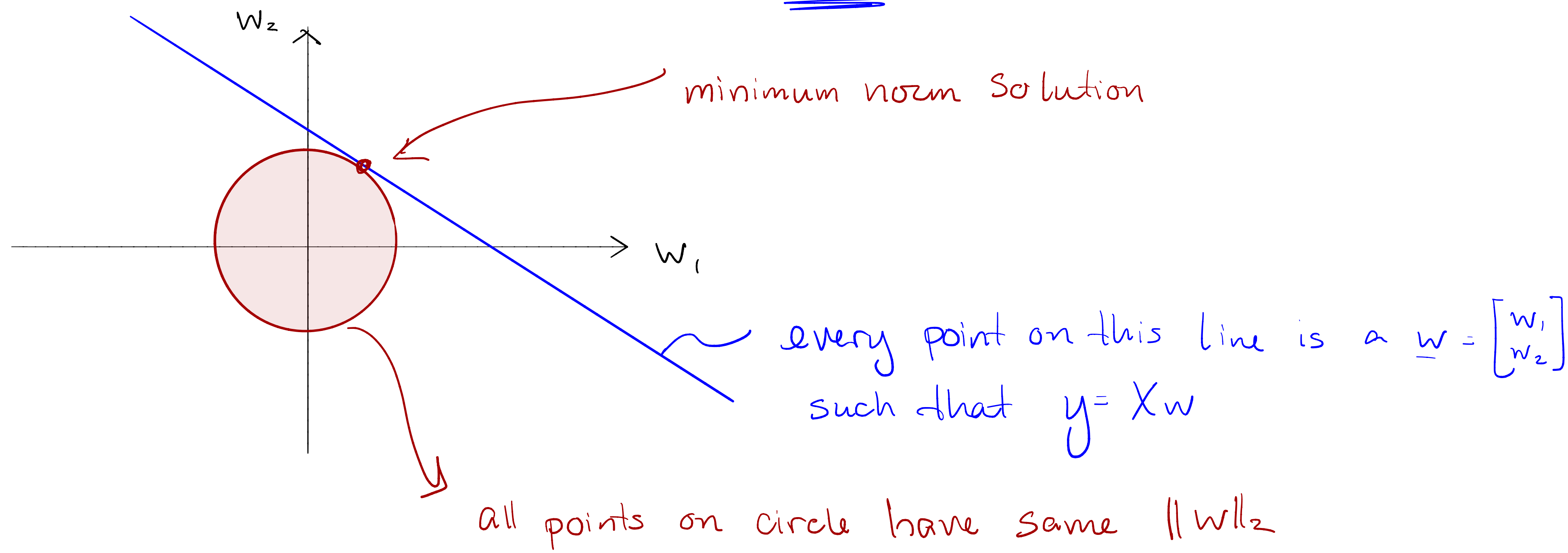
$$\Longrightarrow \quad \underline{\pi}^{(k)} = \frac{A^k \underline{\pi}^{(0)}}{\|A^{(k)} \underline{\pi}^{(0)}\|_2} \to \frac{c_1 \lambda_1^k \underline{v}_1}{\|c_1 \lambda_1^k \underline{v}_1\|} = \underline{v}_1 \quad (\text{or } -\underline{v}_1)$$

Extra Notes on Background

(not for in-class lectures)

We want to find w so that $y = Xw$. Imagine columns of X are linearly dependent $\Longleftrightarrow$ X has some singular values $= 0$. $\Rightarrow$ ∞ many w so that $y = Xw$.

$\Rightarrow$ Minimum norm solution: $\min_w \|w\|_2^2$ such that $y = Xw$ (1)



Connection to Ridge Regression:

If data had a little noise, we might choose $\min_w \|w\|_2^2$ such that $\|y - Xw\| <$ small threshold (2)

Using optimization theory and Lagrange multipliers, we can show that

$$\min_w \|w\|_2^2 + \frac{1}{\lambda}\|y - Xw\|^2 \qquad (3)$$

(3) has the same solution as (2) (for the right choice of λ)

(3) is Ridge regression