MLCheatsheet

1. Bài toán Gốc: Toán học (Theory)

1. Thiết lập:

- $x, w \in \mathbb{R}^{M-1}$; Siêu phẳng: $w^T x + b = 0$.

2. Dẫn dắt: Max Margin $\Leftrightarrow$ Min $\|w\| \Leftrightarrow$ Min $\frac{1}{2}\|w\|^2$.

3. Công thức Primal:

$$\boxed{\begin{aligned} &\min_{w,b} \frac{1}{2}\|w\|^2 \\ &\text{s.t. } t_n(w^T x_n + b) \geq 1, \forall n \end{aligned}}$$

2. Bài toán Gốc: Implementation (CVXOPT)

Solver: $\min \frac{1}{2}u^T P u + q^T u$ s.t $Gu \leq h$. **Biến:** $u = [\mathbf{w}; \mathbf{b}]_{(M\times 1)}$.

Mapping:

$$\boxed{\begin{aligned} &P = \text{diag}(\mathbf{1}, \ldots, \mathbf{1}, \mathbf{0})_{(M\times M)} \\ &q = \mathbf{0}_{(M\times 1)} \\ &G = -[\text{diag}(\mathbf{t}) \cdot \mathbf{X}, \quad \mathbf{t}]_{(N\times M)} \\ &h = -\mathbf{1}_{(N\times 1)} \end{aligned}}$$

3. Lagrangian & Điều kiện KKT

1. Hàm Lagrangian:

$$\mathcal{L} = \frac{1}{2}\|w\|^2 + \sum_{n=1}^{N} \underbrace{\alpha_n}_{\geq 0} \underbrace{\{1 - t_n(w^T x_n + b)\}}_{\substack{\text{đúng: } \leq 0 \\ \text{sai: } > 0}}$$

$$= \frac{1}{2}\|w\|^2 + E_{\text{data}} \text{ (cost trên dữ liệu)}$$

Tính chất: Với $\alpha_n \geq 0$, (w^*, b^*) tối ưu:

$$\boxed{\mathcal{L}(w^*, b^*, \alpha) \leq \frac{1}{2}\|w^*\|^2}$$

2. Tính đạo hàm:

$$\frac{\partial \mathcal{L}}{\partial w} = w - \sum_{n=1}^{N} \alpha_n t_n x_n$$

$$\frac{\partial \mathcal{L}}{\partial b} = -\sum_{n=1}^{N} \alpha_n t_n$$

3. Điều kiện dừng (KKT-1):

- $\nabla_w \mathcal{L} = 0 \Rightarrow \boxed{w = \sum_{n=1}^{N} \alpha_n t_n x_n}$ (1)
- $\nabla_b \mathcal{L} = 0 \Rightarrow \boxed{\sum_{n=1}^{N} \alpha_n t_n = 0}$ (2)

4. Các điều kiện KKT khác:

- **(KKT-2)** Ràng buộc gốc: $1 - t_n(w^T x_n + b) \leq 0$
- **(KKT-3)** Ràng buộc dual: $\alpha_n \geq 0$, $\forall n$
- **(KKT-4)** Điều kiện bù: $\alpha_n\{1 - t_n(w^T x_n + b)\} = 0$

Ý nghĩa: Thỏa KKT $\Rightarrow (w, b, \alpha) = (w^*, b^*, \alpha^*)$

5. Tiêu chuẩn Slater (cho SVM):

Nếu phần trong của tập khả thi không rỗng thì:

$$\exists (w, b) \text{ s.t. } 1 - t_i(w^T x_i + b) < 0, \ \forall i$$

$$\Longrightarrow \text{strong duality } (p^* = d^*) \ \Leftrightarrow \ \text{duality gap} = 0$$

4. Bài toán Dual: Biến đổi (Math)

Thay (1), (2) vào $\mathcal{L}$:

A. Thay w vào $\frac{1}{2}w^T w$:

$$\frac{1}{2}\left(\sum \alpha_i t_i x_i\right)^T \left(\sum \alpha_j t_j x_j\right) = \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j t_i t_j (x_i^T x_j) \ (*)$$

B. Thay w vào $-\sum \alpha_n t_n w^T x_n$:

$$-\sum_n \alpha_n t_n \underbrace{\left(\sum_m \alpha_m t_m x_m\right)^T}_{w^T} x_n = -2 \times (*)$$

C. Kết quả ($A + B = -A$):

$$\boxed{\begin{aligned} \max_{\alpha} g(\alpha) = &\sum_n \alpha_n - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j t_i t_j (x_i^T x_j) \\ \text{s.t.} \quad &\alpha_n \geq 0; \quad \sum \alpha_n t_n = 0 (g(\alpha) \text{ luôn là hàm lồi}) \end{aligned}}$$

5. Bài toán Dual: Implementation (CVXOPT)

Solver: $\min \frac{1}{2}\alpha^T P \alpha + q^T \alpha$ s.t $G\alpha \leq h, A\alpha = b$. **Biến:** $\alpha \in \mathbb{R}^N$.

Mapping:

$$\boxed{\begin{aligned} &P = \mathbf{K}_{\text{Gram}(N\times N)} \quad (K_{ij} = t_i t_j x_i^T x_j) \\ &q = -\mathbf{1}_{(N\times 1)} \quad (\text{Max } \Sigma \to \text{Min } -\Sigma) \\ &G = -I_{(N\times N)}; \ h = \mathbf{0}_{(N\times 1)} \\ &A = t^T; \ b = 0 \end{aligned}}$$

6. Soft Margin & Kernel

1. Soft Margin (ý tưởng): Cho phép một số điểm vi phạm margin bằng biến slack ξ_n, đổi lại bị phạt trong hàm mục tiêu.

2. Bài toán gốc (Primal – soft margin):

$$\min_{w,b,\xi} \frac{1}{2}\|w\|^2 + C\sum_{n=1}^{N} \xi_n$$

$$\text{s.t. } t_n(w^T x_n + b) \geq 1 - \xi_n, \quad \xi_n \geq 0$$

Ý nghĩa C: C lớn $\Rightarrow$ phạt mạnh (ít sai, lề hẹp); C nhỏ $\Rightarrow$ cho sai nhiều (lề rộng).

3. Dạng đối ngẫu (Dual):

$$\mathcal{L} = \frac{1}{2}\|w\|^2 + C\sum_n \xi_n - \sum_n \alpha_n\{t_n(w^T x_n + b) - 1 + \xi_n\} - \sum_n \mu_n \xi_n$$

$$\max_{\alpha} \sum_{n=1}^{N} \alpha_n - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j t_i t_j \, x_i^T x_j$$

$$\text{s.t. } \boxed{0 \leq \alpha_n \leq C}, \quad \sum_{n=1}^{N} \alpha_n t_n = 0$$

Phân loại theo KKT:

- $\alpha_n = 0$: điểm ngoài margin (không support)
- $0 < \alpha_n < C$: support vector trên margin
- $\alpha_n = C$: support vector vi phạm / phân loại sai

4. Kernel Trick: Thay tích vô hướng:

$$x_i^T x_j \ \longrightarrow \ k(x_i, x_j)$$

$$f(x) = \sum_{n\in SV} \alpha_n t_n \, k(x_n, x) + b, \qquad y = \text{sign}(f(x))$$

Deep Learning

Goal: Multiclass classification ($y \in \{1, \ldots, K\}$).

Model (Softmax):

$$p_{ik} = \frac{\exp(\mathbf{w}_k^\top \mathbf{x}_i + b_k)}{\sum_{j=1}^{K} \exp(\mathbf{w}_j^\top \mathbf{x}_i + b_j)}$$

Loss (Categorical Cross-Entropy):

$$\mathcal{L}_{\text{CE}} = -\frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{K} y_{ik} \log p_{ik} = -\frac{1}{n}\sum_{i=1}^{n} \log p_{i,y_i}$$

Prediction:

$$\hat{y}_i = \arg\max_k p_{ik}$$

Notes:

- $y_{ik} \in \{0, 1\}$: nhãn one-hot của mẫu i tại lớp k
- p_{i,y_i}: xác suất dự đoán của lớp đúng của mẫu i

Vectorized form:

- **Vector (single sample):** for $\mathbf{x} \in \mathbb{R}^N$, $\mathbf{W} \in \mathbb{R}^{M\times N}$, $\mathbf{b} \in \mathbb{R}^M$
  $\mathbf{y} = \mathbf{W}\mathbf{x} + \mathbf{b} \in \mathbb{R}^M$
- **Matrix (mini-batch):** for $\mathbf{X} \in \mathbb{R}^{B\times N}$ (rows are samples)
  $\mathbf{Y} = \mathbf{X}\mathbf{W}^\top + \mathbf{1}\mathbf{b}^\top \in \mathbb{R}^{B\times M}$ (broadcast $\mathbf{b}$ to all rows)

Activation functions:

Sigmoid

$$\sigma(x) = \frac{1}{1+e^{-x}}, \qquad \sigma'(x) = \sigma(x)(1 - \sigma(x))$$

Tanh

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}, \qquad \tanh'(x) = 1 - \tanh^2(x)$$

ReLU

$$\text{ReLU}(x) = \max(0, x), \qquad \text{ReLU}'(x) = \begin{cases} 1, & x \geq 0 \\ 0, & x \leq 0 \end{cases}$$

Leaky ReLU

$$\text{LReLU}(x) = \begin{cases} x, & x \geq 0 \\ \alpha x, & x \leq 0 \end{cases}, \qquad \text{LReLU}'(x) = \begin{cases} 1, & x \geq 0 \\ \alpha, & x \leq 0 \end{cases}$$

SiLU (Swish)

$$\text{SiLU}(x) = x\,\sigma(x), \qquad \text{SiLU}'(x) = \sigma(x) + x\,\sigma(x)(1 - \sigma(x))$$

Linear Regression Solution:

Model:

$$\hat{y} = \mathbf{w}^\top \mathbf{x} + b$$

Loss (MSE):

$$\mathcal{L} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 = \frac{1}{n}\|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2$$

Closed-form Solution (Normal Equation):

$$\mathbf{w}^* = (\mathbf{X}^\top \mathbf{X})^{-1}\mathbf{X}^\top \mathbf{y}$$

With regularization (Ridge):

$$\mathbf{w}^* = (\mathbf{X}^\top \mathbf{X} + \lambda \mathbf{I})^{-1}\mathbf{X}^\top \mathbf{y}$$

1. Giới thiệu & Mục tiêu PCA

1. Đầu vào:

- Ma trận dữ liệu X có kích thước $N \times D$ (N mẫu, D chiều). Thường D rất lớn.
- Độ phức tạp tính toán tăng lên: ma trận Kgram, kernel, hiệp phương sai, khoảng cách giữa các điểm dữ liệu, số nơron lớp ẩn tăng. Và cần mô hình phức tạp hơn, điểm dữ liệu lớn hơn dễ gây overfit.

2. Mục tiêu:

- Giảm số chiều $D \to M$ sao cho $M \ll D$.
- Đặc trưng mới không tương quan (uncorrelated).
- Giữ lại thông tin quan trọng nhất (Variance lớn nhất).

2. Kiến thức Toán nền tảng

1. Hình chiếu: Chiếu vector x lên vector u: $l = \frac{u^T x}{\|u\|}$.

2. Dạng bậc hai & Đạo hàm:

- $u = [u_1, u_2, \ldots, u_N]^T$, $v = [v_1, v_2, \ldots, v_M]^T$.
- $A = [a_{ij}]_{N \times M}$.
- $u^T A v = \sum_{i=1}^{N} \sum_{j=1}^{M} a_{ij} u_i^T v_j$. ($aij$ và $u_i^T v_j$ là vô hướng).
- $\frac{\partial u^T A u}{\partial u} = 2Au$ (với A đối xứng).

3. Eigenvectors & Eigenvalues:

$Au = \lambda u \quad \Leftrightarrow \quad (A - \lambda I)u = 0 (3.7)$

$=> \det(A - \lambda I) = 0 (3.8)$

Giải 3.8 để tìm λ_i (có tối đa N eigenvalues), thay vào 3.7 để tìm u_i.

4. Tính chất eigenvectors và eigenvalues S:

- Ma trận A là đối xứng chứa các số thực.
- Có N eigenvector trực giao và N eigenvalue thực không âm.
- Hạng của A bằng số eigenvalue lớn hơn 0.
- Định thức của A là tích các eigenvalue. Trace của A là tổng các eigenvalue.
- Khi hạng của A là N, ma trận A^-1 có N eigenvalue = $1/\lambda_i$. (i = 1,...,N)

3. Bài toán PCA

1. Hai quan điểm về PCA:

- PCA chuyển dữ liệu đầu vào là N điểm thuộc D chiều sang không gian con có kích thước M chiều (M < D) sao cho:
- Khi chiếu dữ liệu lên không gian con này, phương sai của dữ liệu là lớn nhất: $argmax_u \frac{1}{N} \sum_{k=1}^{N} x_{u,k}^2$
- Hoặc khi chiếu dữ liệu lên không gian con này, khoảng cách từ điểm dữ liệu đến không gian con là nhỏ nhất: $argmin_u \sum_{k=1}^{N} d_{u,k}^2$

2. Cực đại hóa phương sai:

- Tìm vector u đầu tiên cho ra phương sai lớn nhất. (Tối ưu có ràng buộc và dùng Lagrangian để giải)
- Giả sử đã tìm ra M vector đầu tiên: Các vector đều có chiều dài 1 và trực giao nhau đôi một.
- Tìm vector thứ M+1 sao cho trực giao với M vector trước và có phương sai lớn nhất.

Công thức tính định thức ma trận

Ma trận 2×2:

$$\begin{vmatrix} a & b \\ c & d \end{vmatrix} = ad - bc$$

Ma trận 3×3 (quy tắc Sarrus):

$$\begin{vmatrix} a & b & c \\ d & e & f \\ g & h & i \end{vmatrix} = aei + bfg + cdh - ceg - bdi - afh$$

a. Khi k = 1:

- Mục tiêu: Tìm u để phương sai của dữ liệu chiếu lên u là lớn nhất.
- Điểm trung tâm của dữ liệu: $\mu = \frac{1}{N} \sum_{n=1}^{N} x_n$. $z_n = x_n - \mu$ là điểm tương ứng của x_n sau khi dịch về μ.
- Variance của dữ liệu chiếu lên u: $\sigma^2 = \frac{1}{N} \sum_{n=1}^{N} (u^T z_n)^2 = \frac{1}{N} \sum_{n=1}^{N} (u^T z_n)(z_n^T u) = \frac{1}{N} \sum_{n=1}^{N} u^T [z_n z_n^T] u = \frac{1}{N} \sum_{n=1}^{N} u^T [(x_n - \mu)(x_n - \mu)^T] u = u^T S u$.
- Covariance matrix (ma trận hiệp phương sai): $S = \frac{1}{N} \sum_{n=1}^{N} (x_n - \mu)(x_n - \mu)^T = Z^T Z = (X - \mu)^T (X - \mu)$. S là ma trận vuông, đối xứng $D \times D$, bán định dương.
- Bài toán tối ưu: $argmax_u u^T S u$ với ràng buộc $u^T u = 1$. (S bán định dương => hàm mục tiêu lồi. Giải bằng pp nhân tử Lagrange).
- Lagrangian: $\mathcal{L}(u, \lambda) = u^T S u - \lambda(u^T u - 1)$. Đạo hàm theo u và đặt bằng 0: $\frac{\partial \mathcal{L}}{\partial u} = 0 \Rightarrow Su - \lambda u = 0 \Rightarrow Su = \lambda u$.
- Bài toán trở thành bài toán tìm eigenvector và eigenvalue của ma trận S: $Su_1 = \lambda_1 u_1$. Nghiệm u_1 là eigenvector cho variance lớn nhất λ_1.
- Tính chất của eigenvalues và eigenvectors của S: $\lambda_1 = u_1^T S u_1$ (phương sai lớn nhất).

b. Khi k = M:

- Bài toán tối ưu: $u_M = argmax_u u^T S u$ với ràng buộc $u^T u = 1$ và $u^T u_k = 0$ với $k = 1, 2, \ldots, (M-1)$.
- Mục tiêu: Tìm vector u_M sao cho trực giao với tất cả các vector trước và khi chiếu dữ liệu lên hướng này có phương sai lớn nhất (Vẫn $\leq \lambda_1, \lambda_2, \ldots, \lambda_{M-1}$). Độ lớn của u_M là 1.
- Giả thiết: Đã tìm được cặp u_k, λ_k với $k = 2, \ldots, M$ từ bài toán tối ưu ở trên.

c. Khi k = M+1:

- Bài toán tối ưu và mục tiêu: Tương tự như trên nhưng thay M bằng M+1 và M-1 bằng M. Cần suy ra bài toán tối ưu ở đây có nghiệm.
- Lagrangian: $\mathcal{L}(u, \lambda, \mu) = u^T S u - \lambda(u^T u - 1) - \sum_{k=1}^{M} \mu_k (u^T u_k)$.
- Đạo hàm theo u: $\frac{\partial \mathcal{L}}{\partial u} = 2Su - 2\lambda u - \sum_{k=1}^{M} \mu_k u_k$.
- Điểm làm lagrangian cực tiểu là nghiệm khi đạo hàm bằng 0: $2Su - 2\lambda u - \sum_{k=1}^{M} \mu_k u_k = 0$.
- Ở phương trình này, $\mu_k = 0$ vì u trực giao với tất cả u_k. Vậy ta nhân hai vế với u_k và có lại phương trình $Su = \lambda u$.

2. Giải thuật thu giảm số chiều:

- Tính ma trận hiệp phương sai và tìm các cặp u_k, λ_k (numpy.linalg.eig, numpy.linalg.eigh).
- Chọn M eigenvector $e_1, e_2, \ldots, e_M$. Chọn M bằng kiểm thử chéo.
- Chiếu dữ liệu gốc lên M vector để thu dữ liệu mới.

3. Singular Value Decomposition (SVD):

- SVD là kĩ thuật tổng quát của PCA.
- PCA: Tính ma trận hiệp phương sai S rồi phân rã eigen trên S.
- SVD: Phân rã SVD trên chính ma trận gốc (X), quá trình phân rã ổn định hơn.
- Ma trận X được SVD phân rã thành $X = USV^T$. Với $U(N \times N)$: có cột là eigenvectors của XX^T. $V(D \times D)$: có cột là eigenvectors của $X^T X$. $S(N \times D)$: ma trận có đường chéo chính là singular values xếp từ lớn đến nhỏ.

Giải thuật

- Tính $Z = X - m^T$ với m là total mean = $\frac{1}{N} \sum_{n=1}^{N} x_n$.
- Dùng SVD phân rã $Z = USV^T$ (numpy.linalg.svd).
- Chọn M vectors đầu tiên của V tương ứng M singular values lớn nhất tạo $\hat{V}$.
- Chiếu dữ liệu Z lên M eigenvectors: $X_{pca} = Z\hat{V}$.

4. Tổng kết

PCA:

- $A = UDU^T = \sum_{i=1}^{r} \lambda_i u_i u_i^T$.
- A: ma trận hiệp phương sai.
- r: hạng của A.
- Thu giảm chiều: Chỉ giữ lại M eigenvector và chiếu X lên các vector này.

SVD:

- $A = USV^T = \sum_{i=1}^{r} s_i u_i v_i^T$.
- A: ma trận bất kỳ.
- r: hạng của A.
- Thu giảm chiều: Chỉ giữ lại M eigenvector và xấp xỉ X lên các vector này.

5. PCA qua API

```
1. import numpy as np
2. from sklearn.decomposition import PCA
3. X = np.array([[-1, -1], [-2, -1], [-3, -2], [1, 1], [2, 1], [3, 2]])
4. pca = PCA(n_components=2)
5. pca.fit(X)
6. print(pca.explained_variance_ratio_)
7. print(pca.singular_values_)
```

1. Giới thiệu về LDA

1. Đầu vào:

- X có kích thước $N \times D$, D rất lớn.
- t_k là nhãn lớp, có C lớp khác nhau.

2. Mục tiêu:

- Giảm số chiều từ D xuống M và $M \leq C - 1$.
- Dữ liệu sau khi giảm có tính phân tách cao nhất.

2. LDA qua API (Scikit-learn)

1. from sklearn import datasets
2. from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
3. iris = datasets.load_iris()
4. X = iris.data
5. y = iris.target
6. target_names = iris.target_names
7. lda = LinearDiscriminantAnalysis(n_components=2)
8. X_r2 = lda.fit(X, y).transform(X)

3. Bài toán tối ưu

1. Các thành phần:

- Tâm của dữ liệu thuộc mỗi lớp: $m_1 = \frac{1}{N_1}\sum_{n\in C_1} x_n$, $m_2 = \frac{1}{N_2}\sum_{n\in C_2} x_n$
- Khoảng cách giữa 2 tâm khi chiếu lên vector w: $d_{12} = w^T S_B w$
- Between-class covariance matrix: $S_B = (m_2 - m_1)(m_1 - m_2)^T$
- Variance của lớp 1 sau khi chiếu lên w: $\sigma_1^2 = w^T S_{W1} w$
- Within-class covariance matrix: $S_{W1} = \sum_{n\in C_1}(x_n - m_1)(x_n - m_1)^T$
- Within-class covariance matrix của cả 2 lớp: $S_W = S_{W1} + S_{W2}$
- Within-class variance: $\sigma^2 = w^T S_W w$

2. Hàm mục tiêu (Quan điểm của Fisher):

Tìm hướng $w^* = \arg\max_w J(w)$ để cực đại hóa tỷ số:

$$\boxed{J(w) = \frac{w^T S_B w}{w^T S_W w}}$$

3. Tìm nghiệm:

- Đạo hàm $\nabla_w J = 0 <=> S_B w = \lambda S_W w$, $\lambda = \frac{w^T S_B w}{w^T S_W w}$
- => Phương trình eigenvector: $S_W^{-1} S_B w = \lambda w$
- Chúng ta chỉ cần hướng của w, nên dùng $S_B w = (m_2 - m_1)(m_2 - m_1)^T w$
- Hướng của vector w là $S_W^{-1}(m_2 - m_1)$; Cần chuẩn hóa w để có độ dài đơn vị.

4. Trường hợp có C lớp

====== Phần này chưa làm xong ======

1. Thiết lập tham số:

- Số lớp: C ($C > 2$).
- Vector trung bình lớp k: m_k.
- Vector trung bình toàn cục: $m = \frac{1}{N}\sum_{k=1}^{C} N_k m_k$.

2. Xây dựng các Ma trận Scatter:

- **Within-class (S_W):** Tổng độ phân tán trong từng lớp.

$$S_W = \sum_{k=1}^{C} S_k$$

  (với S_k là ma trận scatter của lớp k).
- **Between-class (S_B):** Độ phân tán của các tâm lớp so với tâm chung.

$$S_B = \sum_{k=1}^{C} N_k (m_k - m)(m_k - m)^T$$

3. Hàm mục tiêu (Objective Function): Tìm ma trận chiếu W để tối đa hóa tỷ số định thức:

$$\boxed{J(W) = \frac{\det(W^T S_B W)}{\det(W^T S_W W)}}$$

4. Giải pháp và Kết quả:

- Bài toán trị riêng: $S_B w_i = \lambda_i S_W w_i$ (hoặc $S_W^{-1} S_B w_i = \lambda_i w_i$).
- W được tạo thành từ các **eigenvectors** ứng với các **eigenvalues** lớn nhất.
- **Số chiều tối đa:** $K \leq C - 1$ (vì hạng của S_B tối đa là $C - 1$).