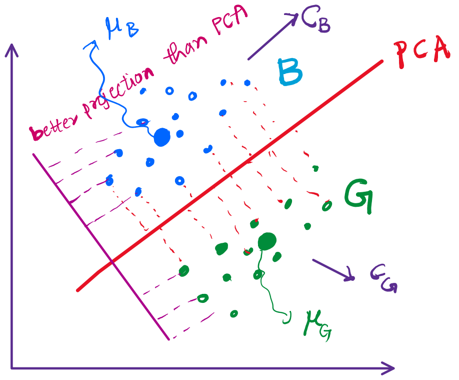
Lecture 18: Dimensionality Reduction (contd.)

22 March 2024 11:03

Recap: Unsupervised PCA: maximize variance of projected data (does not use class information)

Supervised:



Objective is:

i) The mean of the two classes are well separated

ii) The data in the same class are not well separated

$$J(u) = \left| \mu_B^T u - \mu_G^T u \right|$$

We adopt Linear Discriminant Analysis (LDA).

LDA

The variance within a class is called "scatter".

LDA: 2-class

$$S_i = \frac{1}{|C_i|} \sum_{x \in C_i} (x - \mu_i)(x - \mu_i)^T$$

where, C_i : data points belonging to class i

Variance of the projected data points of C_i on u.

$$\sigma_i = u^T S_i u$$ (Refer PCA notes for detailed derivation (Lec 17))

Sum of variances $= \sigma_1 + \sigma_2 = u^T S_1 u + u^T S_2 u = u^T S_W u$

where $S_W = S_1 + S_2$ Covariance within class

We need to maximise the separation b/w classes:-

$$(u^T \mu_1 - u^T \mu_2)^2 = u^T (\mu_1 - \mu_2)(\mu_1 - \mu_2)^T u = u^T S_B u$$

where S_B is covariance b/w classes.

LDA as an optimization problem

$$J(u) = \frac{u^T S_B u}{u^T S_w u}$$

Scale invariant so that u need not be unit vector!

$$\boxed{\max J(u) \Rightarrow \max u^T S_B u \quad st \quad u^T S_W u = 1}$$

Consider the Lagrangian of this $f^{\underline{n}}$:

$$\mathcal{L}(\lambda, u) = -u^T S_B u + \lambda(u^T S_w u - 1)$$

(Assume S_w to be invertible)

we need for minimizing $\mathcal{L}(\lambda, \omega)$, $\frac{\partial \mathcal{L}}{\partial u} = 0$

$$\Rightarrow -2S_B u + 2\lambda S_w u = 0$$

$$\Rightarrow S_w^{-1} S_B u = \lambda u \Rightarrow S_B u = \lambda S_w u$$

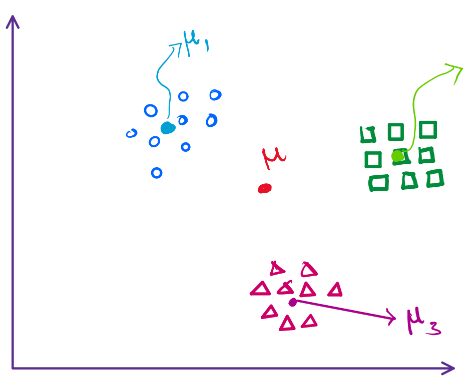
We should project the data points on a direction which is an eigenvector corresponding to maximum eigen value.

$$\boxed{\begin{aligned} &\max \ u^T(\lambda S_w u) = \lambda \\ &st. \ u^T S_w u = 1 \end{aligned}}$$

$$S_W^{-1} S_B = V \Sigma V^T$$

$\Rightarrow v_1$ is direction to project.

LDA: more than 2 classes



$$S_W = S_1 + S_2 + S_3 + \dots + S_c$$

$$S_B = \sum_{i=1}^{c} n_i (\mu_i - \mu)(\mu_i - \mu)^T$$

$|C_i|$ = no. of data points in class i.

$$\mu = \frac{1}{n} \sum_{i=1}^{c} n_i \mu_i$$

Optimization problem:

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

Optimal solution $\Rightarrow$ $S_W^{-1} S_B u = \lambda u$

For projecting into a k-dimensional space $(k \geq c)$, can we just pick the largest k eigenvectors as in PCA?

We can write S_B as:

$$S_B = \overset{\text{column vectors.}}{\left[\sqrt{n_1}(\mu_1 - \mu) \quad \sqrt{n_2}(\mu_2 - \mu) \quad \cdots \quad \sqrt{n_c}(\mu_c - \mu)\right]} \overset{\text{row vectors}}{\begin{bmatrix} \sqrt{n_1}(\mu_1 - \mu)^T \\ \vdots \\ \sqrt{n_c}(\mu_c - \mu)^T \end{bmatrix}}$$

$$= AA^T$$

$$\boxed{\text{Result: } rank(AB) \leq \min\{rank(A), rank(B)\}}$$

Thus S_B doesn't have full rank (a low rank matrix)

$\Rightarrow$ We can find at most $(c-1)$ discriminating directions.

> Observe that:
>
> $\sqrt{n_1}\sqrt{n_1}(\mu_1 - \mu) + \cdots + \sqrt{n_c}\cdot\sqrt{n_c}(\mu_c - \mu)$
>
> $= \sum n_i(\mu_i - \mu) = 0$
>
> $\Rightarrow$ columns of A are linearly dependent

LDA algorithm

1. Compute the means of each class, μ_i
2. Calculate S_W and S_B
3. Find top k non zero eigenvectors corresponding to eigenvalues of $S_W^{-1} S_B$, $k \leq c-1$

   $u_1, u_2, \ldots u_k$
4. Create $U = [u_1, \ldots\ldots u_k]$
5. Project x to $U^T x$.

End Of Machine Learning part Of this course

Artificial Intelligence

	Human Side	Rational Side

	Human Side	Rational Side
Thinking	NLP, vision, automated reasoning, ---- ML	Logician's approach (complete information) can't handle uncertainities
Acting	Cognitive Science - brain's functions	Agent based approach - single agent - multiple agent

Ref: Russell and Norvig.

What is Rationality?

Making decisions with reason.

Depends on:

i) performance measure

ii) agent's prior knowledge about the environment and other agents.

iii) actions available to the agents

iv) history (past states/actions)

Common performance metric

- ML ⟶ loss function (minimize)
- Robotics ⟶ Reinforcement learnining: reward function (maximize)
- Multi-agent systems ⟶ (more than oneagents) utility functions.

Consider a two player game:

A	B	C
-50 50	1 3	-10 20

- You choose one of bins as above
- Opponent chooses a number from that bin

- Your performance measure/utility is the number picked.

If the opponent chooses the number randomly ($p = 1/2$), choose bin with max expected value i.e., C here.

If the opponent is adversarial, i.e., chooses minimum, choose bin B.

Game tree: