00. 두 개 클래스 데이터 생성 및 산점도 시각화

1.데이터 생성

- $\mu_1 = (0, 0)$, $\mu_2 = (0, 5)$를 평균으로 하는 2차원 정규분포에서 두 개 클래스를 생성했습니다.
- 공분산 행렬 \Sigma = \begin{pmatrix}10 & 2 \\ 2 & 1\end{pmatrix} 을 두 클래스가 동일하게 사용합니다.
- 클래스 1, 클래스 2 각각 100개씩 목데이터를 샘플링하여, 총 200개의 2D 데이터를 얻었습니다.

2.산점도 시각화

- 생성된 데이터를 matplotlib를 이용해 2차원 산점도로 표현했습니다.
- 클래스 1의 표본은 파란색 별표(*) 마커로, 클래스 2의 표본은 빨간색 원형(o) 마커로 표시됩니다.
- 축 범위는 x축: [-10, 10], y축: [-5, 10]**으로 제한하여, 그림 1과 유사한 스케일로 분포가 보이도록 했습니다.

3.파이썬 코드

- 구체적인 코드는 00.py 파일을 참조하세요.
- 사용된 주요 라이브러리는 numpy와 matplotlib입니다.

4.결과 산점도

- 클래스 1 → 파란색 별표(*)
- 클래스 2 → 빨간색 원형(o)
- 각 클래스별 100개씩 표시되며, 그래프 상단에는 “Sample Data” 제목, 우측 상단에는 범례를 배치하여 클래스를 구분했습니다.

01. 2차원 데이터에서의 **PCA**와 **LDA** 적용 및 시각화

1.데이터 생성

- 클래스 1: 평균 (0,0), 클래스 2: 평균 (3,3), 공분산은 단위행렬($\begin{pmatrix}1 & 0 \\ 0 & 1\end{pmatrix}$).
- 클래스별로 50개씩, 총 100개의 2차원 점을 만들었습니다.

2.PCA

- 데이터 전체 평균을 구하고, 각각에서 빼서 중심화(Centering) 했습니다.
- 공분산 행렬을 만든 뒤, 고유분해(Eigen Decomposition)하여 가장 큰 고유값에 해당하는 고유벡터를 추출합니다.
- 이 벡터가 첫 번째 주성분(데이터 분산이 가장 큰 방향)입니다.

3.LDA

- 클래스 1, 클래스 2의 평균(μ_1, μ_2)을 각각 구합니다.
- 클래스 내부 산포행렬 S_1, S_2를 구해 $S_w = S_1 + S_2$를 얻고,
- \mathbf{w}_{LDA} = S_W^{-1}(μ_2 - μ_1) 로 LDA 벡터를 계산합니다.
- 이는 “두 클래스를 가장 잘 구분할 수 있는” 방향 벡터입니다.

4.시각화

- 두 클래스 점들을 산점도로 그립니다 (파랑, 빨강).
- 첫 번째 주성분 벡터와 LDA 벡터를 (0,0)에서 출발하는 화살표로 표시합니다(초록, 마젠타).
- 실제로는 데이터 중심(평균)에서부터 그리는 게 의미 있지만, 여기서는 어설프게 원점에서부터 그리는 예시를 보였습니다.
- 결과적으로, PCA 축은 데이터 분산이 큰 방향, LDA 축은 클래스 간 차이가 가장 커지는 방향을 보여줍니다.

02. PCA와 LDA 결과 비교

1.시각적 비교

- PCA 벡터는 전체 데이터의 분산이 가장 큰 방향을 기준으로 하므로, 클래스 정보를 사용하지 않습니다.
- LDA 벡터는 클래스 레이블을 활용하여, 클래스 간 차이를 최대화하는 쪽으로 결정됩니다.

2.차이점

- PCA는 감독(레이블) 없이 단순히 “가장 큰 변동(분산)”에 초점을 맞춘 비지도 방식이고,
- LDA는 주어진 클래스 정보를 사용하는 지도 학습 방식입니다.
- 따라서, 그림에서 PCA 축은 “점들이 가장 퍼지는 방향”, LDA 축은 “클래스 1과 2가 최대한 떨어지는 방향”으로 나타납니다.

3.결론

- PCA는 데이터를 압축(차원 축소) 하는 데 효과적이고,
- LDA는 분류나 클래스 구분이 필요한 경우에 적합합니다.
- 두 벡터 방향이 일치하지 않는 것은 당연하며, 실제로 LDA가 클래스 간격을 더욱 크게 둡니다.

03. COIL20 데이터 PCA & LDA 2차원 특징추출 수행결과보고서

1.COIL20 데이터 기반 2차원 특징추출 시스템

- 데이터 준비
  - 첨부된 HW1_COIL20.mat에서 학습 데이터 X와 클래스 레이블 Y를 불러옵니다.
  - COIL20은 총 20개 객체 각각 다양한 각도로 촬영된 이미지들이 포함된 데이터셋입니다.
  - 문제에서 학습 데이터만 사용하라고 했으므로, X, Y만 이용합니다.
- 목표
  - PCA를 통해 2차원으로 투영하여 시각화와 정보량 95%를 먼저 확보하는 차원까지 PCA 축소 후, LDA로 다시 2차원으로 투영
  - 각 클래스가 서로 다른 마커/색으로 표시되어 구별되도록 2차원 평면에 출력합니다.
- 코드 파일
  - 03.py에서 핵심 로직이 구현되어 있으며,exam_plotmulticlass 함수를 사용해 클래스별로 다른 색으로 산점도를 그려줍니다.

2.PCA (2D) 및 LDA (2D) 적용 과정

- PCA (2차원 직행)
  - 전체 학습 데이터에 대해 공분산 행렬을 구합니다.
  - 고유분해(Eigen Decomposition)를 한 뒤, 가장 큰 고유값 2개에 해당하는 고유벡터로 투영합니다.
  - 이렇게 얻은 2차원 데이터는 데이터 분산을 최대한 유지하는 방식이라, 객체 정보는 반영되지 않았지만 전체 분포 파악에 유리합니다.
- PCA (95% 유지) → LDA (2차원)
  - PCA로 누적분산이 95% 이상이 될 때까지 주성분을 선택하여 차원을 축소합니다.
  - 축소된 데이터에 대해, 각 클래스별로 평균(\mu_c)을 구하고, Within-Class Scatter와 Between-Class Scatter를 계산합니다.
  - $\mathbf{w}$ LDA는 두 행렬($S\ W\ -\ 1\ S\ B$)의 고유분해로 얻으며, 상위 2개 축을 선택해 2차원 공간을 형성합니다.
  - 최종적으로 얻어진 Z_lda2는 클래스 간 차이를 최대로 반영하므로, PCA보다 클래스가 더 분리되어 보이는 경향이 있습니다.
- 시각화
  - exam_plotmulticlass(Z, Y) 함수로, 각 클래스마다 서로 다른 마커·색을 할당해 줍니다.
  - 2차원 평면에 20개 객체가 다른 기호(원, 사각형, 별 등)와 색(파랑, 빨강, 초록 등)으로 표시되어 구별됩니다.

3.결과 관찰

- PCA (2D) 결과: 데이터 전체 분산을 기준으로 한 좌표계이므로, 클래스별 색/마커가 섞여 있을 수 있으나, 전체 객체들이 어떻게 퍼져 있는지 한눈에 확인할 수 있습니다.
- LDA (2D) 결과: 클래스 레이블을 이용해 각 객체 간 간격을 더 명확히 구분해주며, 20개 클래스 중 일부가 어느 정도 겹치더라도 비교적 분리가 잘 되는 편입니다.

4.결론

- 정보보존율 95%로 PCA를 우선 수행 후, LDA로 2차원 특징추출을 수행.
- 시각적으로 각 클래스가 서로 다른색으로 출력
- PCA는 비지도 학습, LDA는 지도 학습이므로, 두 결과를 비교함으로써 데이터 특성을 파악하는 데 도움이 됩니다.