01 차원축소 개요

차원 축소는 많은 피처를 가진 데이터의 차원을 줄여 **핵심 정보만 남기고 불필요한 정보를 제거하거나 압축하는 기법**이다. 차원이 많을수록 데이터가 희소해지고 연산량과 과적합 위험이 증가한다. 또한 피처 간 상관관계가 높으면 선형 모델에서 **다중 공선성** 문제가 발생해 성능이 떨어질 수 있다. 따라서 차원을 줄이면 **모델 성능 향상**, **학습 속도 증가**, **시각화 용이성**(2D/3D 표현 가능)이라는 장점이 있다.

차원 축소 방식은 크게 두 가지이다.

- **피처 선택:** 영향이 적은 피처를 제거하고 중요한 피처만 남김
- **피처 추출**: 기존 피처를 조합해 새로운 저차원 특성 생성(기존 값과 다름)

피처 추출 기법은 데이터 속에 숨어 있는 **잠재 요인**을 찾아내는 것이 핵심이다. 예를 들어 학생 평가 데이터에서 여러 점수를 종합해 *학업 능력/소통 능력*과 같은 요약 특성을 만들 수 있다.

대표적 알고리즘:

- **PCA(주성분 분석)**: 분산이 가장 큰 방향으로 축을 새로 만들어 데이터 압축
- **LDA(선형판별분석)**: 클래스 간 거리 최대화, 분류 목적
- **SVD(특이값 분해)**: 행렬 분해 기반, 이미지·텍스트 잠재 의미 추출
- **NMF(비음수 행렬 분해)**: 음수가 없는 데이터에서 부분을 조합해 의미 파악

특히 이미지·텍스트 데이터는 원본 차원이 매우 크기 때문에 차원 축소를 통해 **특징을 요약하고 노이즈를 제거**하면 오히려 예측 성능이 향상된다. 이미지에서는 공통 패턴을 추출해 압축하고 텍스트에서는 단어 패턴으로 **숨겨진 주제**를 찾아낼 수 있다. 결국 차원 축소는 단순한 데이터 압축이 아니라 **데이터의 본질을 요약하여 해석력과 모델 성능을 높이는 과정**이다.

02. PCA

PCA(주성분 분석): 고차원 데이터를 더 적은 차원으로 축소하면서 **데이터의 중요한 정보를 최대한 보존**하는 대표적인 차원 축소 기법

기존 데이터의 피처들 사이의 분산과 상관관계를 이용하여 데이터를 가장 잘 설명할 수 있는 새로운 축(주성분)을 찾아낸다.

왜 PCA를 쓰는가?

문제	PCA로 해결되는 점
차원이 너무 많으면 거리 계산이 어려워짐	낮은 차원의 공간으로 투영하여 학습 안정성 증가
피처 간 상관관계가 높으면 모델 성능 저하 (다중공선성)	상관관계 높은 변수들을 새 축에 통합
시각화 어려움	2D/3D로 축소하여 패턴 시각화 가능
학습 시간 증가	차원 축소로 연산 시간 감소

PCA의 동작 원리

1. 입력 데이터의 **공분산 행렬** 계산
2. 공분산 행렬에서 고유값과 **고유벡터**추출
3. 가장 큰 고유값을 가진 벡터부터 정렬 → **주성분 선택**
4. 선택된 주성분으로 원본 데이터를 **선형 변환**

공분산 행렬과 고유벡터

- **공분산 행렬:** 변수들 간의 관계(상관성)를 나타내는 정방행렬
- **고유벡터**: 공분산 행렬이 데이터를 가장 잘 퍼지게 하는 방향
- **고유값**: 해당 방향이 가진 데이터 분산

PCA 적용 예: Iris 데이터

03. LDA

LDA(선형 판별 분석): 지도학습 기반 차원 축소 기법

클래스 간 분리를 최대로 하는 방향으로 데이터를 투영해 차원을 축소. PCA가 데이터의 분산(변동성)이 가장 큰 방향을 찾는 반면, LDA는 **클래스를 가장 잘 구분할 수 있는 축**을 찾음.

LDA는 두 가지 분산 고려

- **클래스 간 분산**: 서로 다른 클래스 중심 간 거리. → 크게 할수록 좋음
- **클래스 내 분산**: 같은 클래스 안의 데이터 퍼짐 정도. → 작게 할수록 좋음

LDA 수행 절차:

1. 각 클래스의 평균 벡터 계산
2. 클래스 내 분산 행렬 S_W및 클래스 간 분산 행렬 S_B계산
3. $S_W^{-1}S_B$행렬의 고유값과 고유벡터 계산
4. 가장 큰 고유값에 대응하는 고유벡터를 선택해 데이터 변환

LDA는 분류 문제에 유용하며 특히 클래스 구분이 명확한 데이터에서 성능이 좋다. PCA와 달리 **레이블 정보가 반드시 필요**하다. 예를 들어 붓꽃 데이터셋에 LDA를 적용하면 클래스가 명확히 구분되도록 차원이 축소되며, 결과는 PCA의 2차원 결과와 시각적으로 유사하지만 **분류 목적에 더 최적화**된 형태가 나온다.

04. SVD(Singular Value Decomposition)

SVD(특이값 분해): 행렬을 세 개의 행렬로 분해하는 기법
행렬 A를 SVD로 분해: $A = U\Sigma V^T$

- U: 입력 행렬의 **왼쪽 특이벡터**(열 직교), 데이터가 projected 되는 방향
- Σ: 특이값을 대각 성분으로 가지는 **대각행렬**, 각 축의 중요도(에너지) 의미
- V^T: **오른쪽 특이벡터**로 원본 feature 공간 방향

특이값이 클수록 해당 축이 데이터를 더 잘 설명하며 작은 특이값은 노이즈에 가까운 정보임을 의미

SVD는 **밀집행렬**뿐만 아니라 **희소행렬에도 적용가능.** 데이터를 표준화하여 PCA와 Truncated SVD 결과를 비교하면 동일한 결과를 얻는다.

Truncated SVD: 대규모 데이터에서 큰 특이값 몇 개만 추출하는 방식.

전체 복원이 불가능하지만 정보 손실을 최소화하며 차원을 줄일 수 있다. 특히 텍스트 분석(LSA)에서 문서-단어 행렬과 같은 희소 행렬 처리에 많이 사용된다.

SVD는 이미지 압축에도 활용된다. 큰 특이값만 남기면 원본 구조는 유지하면서 데이터 크기를 크게 줄일 수 있기 때문에 고해상도 이미지 저장 및 전송에 효율적이다.
>> SVD는 **데이터의 중요한 구조만 남기고 불필요한 정보를 제거해 차원을 줄이면서도**

의미 있는 패턴을 유지하는 기법

05. NMF (Non-Negative Matrix Factorization)

NMF(비음수 행렬 분해): 모든 값이 0 이상인 행렬을 두 개의 더 작은 **비음수 행렬**로 분해하여 원래 행렬을 근사하는 차원 축소 기법

특히 텍스트 데이터나 이미지처럼 **음수가 의미 없는 데이터**에 잘 맞는다.

예를 들어, 원본 행렬 V를 두 행렬 W와 H로 나누어 ""$V \approx WH$"" 형태로 표현

- W: 원본 데이터의 **잠재 특징**
- H: 각 잠재 특징이 원본 데이터의 **열(특성)을 구성하는 비율**

이 방식은 복잡한 정보 속 잠재 요소를 찾아내는 데 강하다. 단, NMF는 모든 값이 **0 이상이어야 하므로 음수값이 포함된 데이터는 사용 전 변환이 필요**하다.

NMF를 사용하는 이유

- 고차원 데이터를 **의미 있는 낮은 차원 특징 공간**으로 변환
- 이미지·텍스트에서 **패턴, 주제**추출
- 해석 가능성이 높음
  → 음수 없이 "얼마나 기여했는지" 비율로 표현

NMF 활용 분야

분야	**활용 예**
이미지 처리	이미지 압축, 패턴 분석
문서 분석	토픽 모델링 (문서 → 주제 분해)
추천 시스템	유저-상품 평점행렬을 분해해 잠재 요인 추출
클러스터링	비지도 학습 기반 군집화