

10주차 발표자료 요약본

1. 차원 축소개요

- 차원 축소: 매우 많은 피처로 구성된 다차원 데이터 세트의 차원을 축소해 새로운 차원의 데이터 세트를 생성하는 것
 - 다차원의 피처를 차원 축소하면 더 직관적으로 데이터를 해석할 수 있음
 - 피처 선택: 특정 피처에 종속성이 강한 불필요한 피처는 아예 제거하고 데이터의 특징을 잘 나타내는 주요 피처만 선택하는 것
 - 피처 추출: 기존 피처를 단순 압축이 아닌, 피처를 함축적으로 더 잘 설명할 수 있는 또 다른 공간으로 매핑해 저차원의 중요 피처로 압출해서 추출
 - 새롭게 추출된 중요 특성은 기존의 피처가 압축된 것이므로 기존의 피처와는 완전히 다른값이 됨
 - 차원의 저주: 데이터의 차원이 늘어나면 데이터 분석이나 학습이 점점 어려워지는 현상
 - 고차원 데이터셋은 대부분의 훈련 데이터가 서로 멀리 떨어져 있음
 - 새로운 샘플도 훈련 샘플과 멀리 떨어져 있을 가능성 높음.
- ⇒ 해결 방법:
1. 투영:
 - a. 모든 샘플을 부분 공간에 수직으로 투영
 - b. 스위스를 데이터셋의 경우 펼쳐서 2D 데이터셋 얻어야 함
 2. 매니폴드 학습:
 - 매니폴드: 국부적으로 d 차원 초평면으로 보일 수 있는 n 차원 공간의 일부 ($d < n$)
 - 훈련 샘플이 놓여있는 매니폴드를 모델링
 1. 대부분 실제 고차원 데이터셋이 더 낮은 저차원 매니폴드에 가깝게 놓여있다는 매니폴드 가정에 근거
 2. 암묵적으로 분류나 회귀 등이 저차원의 매니폴드 공간에 표현되면 더 간단해질 것이라 가정에 근거 ← 항상 유효하진 않음

2. PCA

- PCA: 여러 변수 간에 존재하는 상관관계를 이용해 이를 대표하는 주성분을 추출해 차원을 축소하는 기법

- PCA의 주성분: 가장 높은 분산을 가지는 데이터의 축을 찾아 이 축으로 차원을 축소하는데, 이것이 PCA의 주성분이 됨
- PCA 차원 축소 방법:
 1. 가장 큰 데이터 변동성(Variance)을 기반으로 첫 번째 벡터 축을 생성
 2. 두 번째 축은 이 벡터 축에 직각이 되는 벡터(직교 벡터)를 축으로 함
 3. 세 번째 축은 다시 두 번째 축과 직각이 되는 벡터를 설정하는 방식으로 축을 생성
 4. 이렇게 생성된 벡터 축에 원본 데이터를 투영하면 벡터 축의 개수만큼의 차원으로 원본 데이터가 차원 축소
- PCA의 선형대수 관점:

입력 데이터의 공분산 행렬을 고윳값 분해하고, 이렇게 구한 고유벡터에 입력 데이터를 선형 변환하는 것

 - 고유벡터: PCA의 주성분 벡터로서 입력 데이터의 분산이 큰 방향을 나타냄
 - 고윳값: 바로 이 고유벡터의 크기를 나타내며, 동시에 입력 데이터의 분산을 나타냄
- 적절한 차원 수 선택:

충분한 분산(예: 95%)이 될 때까지 더해야 할 차원수를 선택.

→ 축소할 차원 수를 임의로 정하지X

 - 방법 1: 충분한 분산을 유지하는 데 필요한 차원수 계산
 - PCA를 fit한후 `pca.explained_variance_ratio_`의 `cumsum`이 95% 이상으로 유지하는데 필요한 차원 수 계산
 - `n_components`에 보존하려는 분산의 비율(0.0~1.0)을 설정하면 자동으로 주성분 개수 결정
 - 방법 2: 설명된 분산을 차원 수에 대한 함수로 그리기
그래프의 변곡점(=Elbow)에서 차원을 축소

3. LDA

- LDA: 선형 판별 분석법
 - PCA와 유사점: 입력 데이터 세트를 저차원 공간에 투영해 차원을 축소하는 기법임
 - PCA와 차이점: LDA는 지도학습의 분류에서 사용하기 쉽도록 개별 클래스를 분별 할 수 있는 기준을 최대한 유지하면서 차원을 축소. PCA는 입력 데이터의 변동성이

가장 큰 축을 찾았지만, LDA는 입력 데이터의 등장 각 클래스를 최대한으로 분리할 수 있는 축을 찾음

- LDA는 특정 공간상에서 클래스를 최대한으로 분리하는 축을 찾기 위해 클래스 간 분산과 클래스 내부 분산의 비율을 최대화하는 방식으로 차원 축소
- LDA 차원 축소 방법:
 1. 클래스 내부와 클래스 간 분산 행렬을 구함. 이 두 개의 행렬은 입력 데이터의 결정 값 클래스별로 개별 피처의 평균 벡터를 기반으로 구함.
 2. 클래스 내부 분산 행렬을 S_w , 클래스 간 분산 행렬을 S_b 라고 하면 다음 식으로 두 행렬을 고유벡터로 분해

$$S_w^T S_b = \begin{bmatrix} e_1 & \cdots & e_n \end{bmatrix} \begin{bmatrix} \lambda_1 & \cdots & 0 \\ \cdots & \cdots & \cdots \\ 0 & \cdots & \lambda_n \end{bmatrix} \begin{bmatrix} e_1^T \\ \cdots \\ e_n^T \end{bmatrix}$$

3. 고유값이 가장 큰 순으로 K개(LDA변환 차수만큼) 추출
4. 고유값이 가장 큰 순으로 추출된 고유벡터를 이용해 새롭게 입력 데이터를 변환

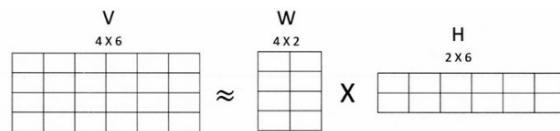
4. SVD

- SVD:
 - PCA와 유사한 행렬 분해 기법 이용
 - 정방행렬만 고유벡터로 분해할 수 있는 PCA와 달리 SVD는 정방행렬뿐만 아니라 $m \times n$ 크기의 행렬 특이값 분해
 - 일반적으로 Σ 의 비대각인 부분과 대각원소 중 특이값 0인 부분 모두 제거하고 제거 된 Σ 에 대응되는 U 와 V 원소도 함께 제거해 차원 줄인 형태로 SVD 적용
- Truncated SVD: Σ 의 대각 원소 중 상위 몇 개만 추출해 더욱 차원 줄인 형태로 분해하는 것
 - 인위적으로 더 작은 차원으로 분해하기 때문에 원본 행렬 정확하게 다시 원복 X
 - BUT 데이터 정보가 압축되어 분해됨에도 불구하고 상당한 수준으로 원복 행렬 근사 가능
 - 넘파이 아닌 사이파이에서만 지원
 - 희소행렬로만 지원돼서 `scipy.linalg.svd` 말고 `scipy.sparse.linalg.svds` 이용해야 함

5. NMF

- NMF:

- Truncated SVD와 같이 낮은 랭크 통한 행렬 근사(Low-Rank Approximation) 방식의 변형
- 원본 행렬 내의 모든 원소 값이 모두 양수라는 게 보장되면 더 간단하게 두 개의 기반 양수 행렬로 분해될 수 있는 기법
- 목적: 공통 특성만을 갖고 정보 줄이는 것

$$\begin{matrix} V \\ 4 \times 6 \end{matrix} \approx \begin{matrix} W \\ 4 \times 2 \end{matrix} \times \begin{matrix} H \\ 2 \times 6 \end{matrix}$$


- W 행렬: 행렬 R과 같은 인덱스의 행에 특성이 얼마나 적합한지 판단하는 가중치 나타내는 Latent representation 행렬
- H 행렬: 행렬 R과 같은 열이 특성에 얼마나 중요한지 나타내는 Latent features 행렬
- 행렬 W, H에 값이 채워지는 방식:
 - 행렬 W, H에 초기값 무작위로 설정하고 행렬 R과 W^*H 간의 거리 최소화하는 방향으로 값 갱신
 - 거리 최소화하기 위해 사용하는 거리함수나 목적함수에 따라 값 달라짐
⇒ 중배 갱신 규칙 (Multiplicative Update Rules)
 1. 네 개의 갱신 행렬 생성
 2. 행렬 W 갱신하기 위해 행렬 W의 모든 값을 식 (a) 내의 대응하는 값과 곱하고 식 (b) 내의 대응하는 값으로 나눔
 3. 행렬 H를 갱신하기 위해 행렬 H 내의 모든 값을 식 (c) 내의 대응하는 값과 곱하고 식 (d) 내의 대응하는 값으로 나눔
 4. 위 과정을 행렬 R과 행렬 W^*H 의 차이가 0이 될 때까지 반복
 - (a) $WN = R \times W^T$
 - (b) $WD = H \times W \times W^T$
 - (c) $HN = H^T \times R$
 - (d) $HD = H^T \times H \times W$
- PCA vs NMF

- PCA:
공분산 기반 고유벡터로 차원을 줄이는 방법이지만, 항상 직교축을 사용해 실제 데이터 구조를 잘 반영하지 못할 수 있음.
- NMF:
데이터를 의미 있는 양수 성분으로 나누어 효율적으로 표현하는 차원 축소 방법.

6. 다른 축소 차원 기법

1) 랜덤 투영

- 개념: 랜덤한 선형 투영을 사용해 고차원 데이터를 저차원으로 변환
- 핵심 아이디어: 무작위 행렬로 변환해도 거리 정보가 잘 보존됨 (존슨-린덴스트라우스 정리)
- 특징:
 - 계산 빠름, 단순함
 - 데이터 수나 차원 수에 의존도 낮음
- 활용 패키지: `sklearn.random_projection`

2) 다차원 스케일링 (MDS, Multidimensional Scaling)

- 개념: 샘플 간의 거리(distance)를 최대한 보존하면서 차원 축소
- 특징:
 - 거리 기반 시각화에 적합
 - 계산 복잡도 높음: 대규모 데이터에는 비효율적

3) Isomap

- 개념: 각 샘플을 가까운 이웃과 연결해 그래프를 만들고, 그 그래프상의 지오데식 거리 (geodesic distance)를 유지하면서 차원 축소
- 특징:
 - 데이터의 비선형 구조(매니폴드)를 보존
 - 전형적인 MDS보다 곡면형 데이터(예: 스위스 롤)에 강함

4) t-SNE (t-distributed Stochastic Neighbor Embedding)

- 개념: 비슷한 샘플은 가깝게, 다른 샘플은 멀리 배치되도록 확률적 방식으로 임베딩

- 특징:
 - 비선형 구조 시각화에 매우 강력
 - 고차원 데이터의 군집 구조 시각화에 자주 사용 (예: MNIST)
 - 계산량 많고, 하이퍼파라미터에 민감

5) 선형 판별 분석 (LDA, Linear Discriminant Analysis)

- 개념: 클래스 간 분산을 최대화하고 클래스 내 분산을 최소화하는 축을 학습
- 특징:
 - 실제는 분류 알고리즘이지만 차원 축소에도 사용
 - 클래스 간 분리도를 높이는 투영면 생성
 - 다른 분류기(SVM 등) 적용 전 전처리용으로 적합

7. 캐글 노트북: Dimensionality Reduction for Beginners

기법	핵심 원리 (2D 변환 기준)	유방암 데이터 시각화 결과	특징 및 장단점
PCA	분산(Variance) 보존	그룹이 분리되나, 일부 겹침. 전반적인 분포를 보여줌.	장점: 빠르다, 축(PC1, PC2)이 해석 가능(히트맵). 한계: 비선형/복잡한 구조에 약함.
MDS	거리(Distance) 보존	그룹이 잘 분리됨. PCA와 다른 형태의 분포.	장점: 샘플 간의 '상대적 거리'를 보존, 비선형 구조 가능. 한계: PCA보다 느릴 수 있음, 축 해석이 어려움.
t-SNE	이웃(Neighbor) 보존	가장 명확한 군집으로 분리. 경계가 뚜렷함.	장점: 군집 시각화에 매우 강력, 복잡한 비선형 구조에 특화. 한계: 계산량이 많음, 축/군집 간 거리 해석 주의.

- 목적에 따라 가장 적합한 기법이 다르다.
 - PCA: 가장 빠르고, 데이터의 전반적인 구조와 주요 성분을 확인할 때
 - MDS: 샘플 간의 상대적인 거리 관계가 중요할 때
 - t-SNE: 데이터의 숨겨진 군집을 시각적으로 명확하게 확인하고 싶을 때