3장. 머신러닝 기초 개념 정리 (딥러닝 파이토치 교과서 기반)

3.1 지도학습 (Supervised Learning)

정의

- 입력 데이터와 그에 대응하는 정답(Label)을 함께 제공하여 모델이 학습하도록 하는 방식.
- 예측 대상이 정해져 있으므로, 정답이 주어진 상황에서 예측 정확도를 높이는 것이 핵심 목표.
- 주로 **분류(Classification)** 와 **회귀(Regression)** 문제에 사용됨.

3.1.1 k-최근접 이웃 (K-Nearest Neighbors, KNN)

개념

- 새로운 입력 데이터가 주어졌을 때, 학습 데이터 중 가장 가까운 K개의 데이터를 찾고, 그 이웃들의 레이블을 참고하여 예측을 수행하는 방식.
- 비모수적(non-parametric)이며, 별도의 학습 과정이 필요하지 않음.

작동 방식

1. 입력 데이터와 기존 학습 데이터 간의 거리를 측정 (보통 유클리디안 거리)
2. 가장 가까운 K개의 데이터를 선택
3. 선택된 이웃들의 다수결 혹은 평균 등을 통해 레이블 예측

장점

- 구현이 매우 간단하고 직관적
- 학습 과정이 없기 때문에 빠르게 적용 가능

단점

- 예측 시 전체 학습 데이터를 탐색해야 하므로 속도가 느림
- 고차원 데이터에서는 거리 개념이 무의미해져 성능 저하 (차원의 저주)
- 이상치에 민감함

적용 팁

- **거리 척도 선택**: 유클리디안 거리 외에도 코사인 거리, 맨해튼 거리 등을 실험하여 가장 성능이 좋은 척도를 선택할 수 있음.
- **데이터 정규화**: 거리 기반 모델이므로, 특성값의 범위가 다르면 성능에 영향을 줌 → 반드시 `StandardScaler` 또는 `MinMaxScaler` 사용.
- **K 값 튜닝**: K가 너무 작으면 과적합, 너무 크면 과소적합 가능 → 교차 검증을 통해 최적의 K 값 선택.
- **차원 축소 병행**: 고차원 데이터일 경우, PCA 등을 활용해 차원을 축소한 후 KNN을 적용하면 성능 향상 가능.

3.1.2 서포트 벡터 머신 (Support Vector Machine, SVM)

개념

- 데이터를 분류할 수 있는 **최적의 결정 경계(Hyperplane)** 를 찾아 마진(Margin)을 최대화하는 방식
- 커널 트릭(Kernel Trick)을 이용하여 비선형 분류도 가능함

작동 방식

1. 분류 경계를 정의하는 결정 초평면(Hyperplane)을 찾음
2. 클래스 간 마진을 최대화하는 방향으로 초평면을 조정
3. 커널 함수를 사용해 고차원 공간으로 변환 가능 (선형 분리가 어려운 경우)

장점

- 고차원 공간에서 강력한 분류 성능
- 일반화 성능이 우수함

단점

- 데이터 양이 많거나 특성이 복잡하면 학습 시간 증가
- 파라미터(C, kernel, gamma 등) 튜닝이 필요함
- 이상치(outlier)에 민감할 수 있음

커널 함수란?

- 선형적으로 분리되지 않는 데이터를 고차원 공간으로 사상하여 분리 가능하게 만드는 함수
- 데이터를 고차원 특징 공간으로 매핑하는 대신, **커널 함수로 두 벡터 간 내적을 계산**함으로써 효율적으로 학습 가능

- 즉, 커널 함수는 실제로 고차원 공간을 계산하지 않고도 고차원 효과를 얻을 수 있도록 도와주는 수학적 트릭

대표적인 커널 함수 종류

1. **선형 커널 (Linear Kernel)**
   - 특징: 기본적인 커널, 고차원 특징이 필요 없는 경우 사용
   - 장점: 빠르고 간단, 해석 가능
   - 단점: 비선형 데이터에는 부적합
2. **다항 커널 (Polynomial Kernel)**
   - 파라미터: 차수 , 상수
   - 특징: 비선형 경계를 모델링할 수 있으나, 고차원 다항식은 계산 비용이 큼
   - 장점: 선형보다 복잡한 패턴 가능
   - 단점: 차수 증가 시 과적합 위험, 계산 복잡도 증가
3. **RBF 커널 (Radial Basis Function, aka Gaussian Kernel)**
   - 파라미터: (거리의 민감도)
   - 특징: 대부분의 상황에서 잘 작동하는 기본값으로 많이 사용됨
   - 장점: 복잡한 비선형 경계도 유연하게 표현 가능
   - 단점: 파라미터 튜닝 필요 (와 C 간 상호작용)
4. **시그모이드 커널 (Sigmoid Kernel)**
   - 신경망의 활성화 함수에서 유래됨
   - 장점: 이론적으로 의미 있음 (신경망과 연결), 일부 상황에선 잘 작동
   - 단점: 다른 커널보다 실전에서는 잘 사용되지 않음

커널 선택 팁

- **데이터가 선형에 가까움**: Linear Kernel 사용
- **복잡한 경계 필요 / 분포가 비선형적**: RBF 커널 권장
- **특성 수가 적고 비선형 정도가 낮음**: Polynomial Kernel도 고려 가능
- 실험적으로 여러 커널을 비교해보고 교차 검증으로 선택하는 것이 중요

작동 방식

1. 분류 경계를 정의하는 결정 초평면(Hyperplane)을 찾음
2. 클래스 간 마진을 최대화하는 방향으로 초평면을 조정
3. 커널 함수를 사용해 고차원 공간으로 변환 가능 (선형 분리가 어려운 경우)

적용 팁

- **커널 함수 선택**: 기본적으로 `linear`를 먼저 시도하고, 성능이 낮으면 `rbf`, `poly`, `sigmoid` 등을 교차 검증으로 실험.
- `C` **파라미터 조정**: 마진 오류 허용도 → 작게 설정하면 마진을 우선시하고, 크게 설정하면 학습 데이터에 더 집중함.
- `gamma` **파라미터**: RBF 커널 사용 시 결정 경계를 얼마나 유연하게 할 것인지 조정하는 하이퍼파라미터 → 작으면 부드러운 경계, 크면 복잡한 경계.
- **데이터 정규화 필수**: 입력 특성의 스케일이 다를 경우, 학습 성능에 큰 영향을 미침 → `StandardScaler` 권장.
- **클래스 불균형 처리**: `class_weight='balanced'` 옵션으로 자동 조정 가능.

3.1.3 결정 트리 (Decision Tree)

개념

- 데이터를 특정 기준에 따라 분할하면서 트리 형태로 예측을 수행하는 모델
- 조건에 따라 데이터를 if-else 방식으로 나누며 예측 수행

작동 방식

1. 가장 정보 이득(Information Gain)이 큰 특성을 기준으로 분할
2. 각 분할에서 반복적으로 자식 노드를 구성하며 리프 노드까지 분기
3. 최종 리프 노드의 다수 클래스 또는 평균으로 예측

장점

- 해석력이 뛰어나고 직관적인 구조
- 범주형/연속형 모두 처리 가능

단점

- 과적합(overfitting)에 매우 취약함
- 데이터에 약간의 변화만 있어도 트리 구조가 크게 바뀔 수 있음 (불안정)

적용 팁

- **트리 가지치기**: 사전 가지치기(`max_depth`, `min_samples_split`, `min_samples_leaf`) 또는 사후 가지치기로 복잡도 제어.
- **앙상블 기법 적용**: Random Forest나 Gradient Boosting을 적용하여 결정 트리의 단점을 보완.
- **데이터 불균형 대응**: 클래스 가중치 조절 혹은 샘플링 기법 사용.

- **특성 중요도 시각화**: `feature_importances_` 속성으로 모델 해석 가능.
- **결측값 처리**: 트리 모델은 결측값을 잘 처리하지 못하므로, 사전에 전처리 필요.

3.1.4 로지스틱 회귀 / 선형 회귀

선형 회귀 (Linear Regression)

- 입력값들의 선형 조합을 통해 연속적인 출력값을 예측하는 모델
- 예: 집값 예측, 판매량 예측 등

로지스틱 회귀 (Logistic Regression)

- 분류 문제를 위한 회귀 모델
- 선형 회귀 결과를 시그모이드 함수에 통과시켜 확률값(0~1)을 출력함
- 이진 분류(예/아니오) 또는 softmax를 통해 다중 분류에도 사용 가능

장점

- 간단하고 학습 빠름
- 출력 해석이 용이

단점

- 복잡한 비선형 분포 분류에 어려움
- 선형 관계를 가정해야 함

적용 팁

- **정규화**: L1(Lasso), L2(Ridge) 정규화를 통해 과적합 방지 → `penalty='l1' or 'l2'` 지정.
- **다중 클래스 분류**: `multi_class='multinomial'` 설정과 함께 `solver='lbfgs'` 사용.
- **피처 스케일링**: 경사 하강법 기반 솔버에서는 정규화 필수 → `StandardScaler` 적용.
- **클래스 불균형 대응**: `class_weight='balanced'` 옵션 사용 가능.

3.2 비지도 학습 (Unsupervised Learning)

정의

- 정답(Label) 없이 데이터의 구조를 이해하려는 학습 방법
- 군집화(Clustering), 차원 축소(Dimensionality Reduction) 등에 활용됨

3.2.1 k-평균 군집화 (K-Means Clustering)

개념

- 데이터를 K개의 군집으로 나누는 알고리즘
- 각 군집은 중심점(centroid)을 가짐

작동 방식

1. 임의로 K개의 중심점을 설정
2. 각 데이터를 가장 가까운 중심점에 할당
3. 군집이 구성되면 각 군집의 평균값으로 중심점을 갱신
4. 더 이상 변화가 없을 때까지 반복

장점

- 계산이 빠르고 구현이 간단함
- 대용량 데이터에도 적용 가능

단점

- 군집 수 K를 미리 정해야 함
- 이상치와 군집 크기 불균형에 민감함

적용 팁

- **K 결정**: Elbow Method, Silhouette Score, Davies-Bouldin Index 등을 활용하여 최적의 K 선택.
- **스케일링**: 중심점을 기준으로 거리 계산하므로, 특성 정규화(`StandardScaler` , `MinMaxScaler`) 필수.
- **초기 중심점 설정**: `k-means++` 알고리즘을 사용하면 초기 중심점 선택에 따른 군집 품질 향상.
- **다중 실행**: 로컬 최소값에 빠지지 않도록 여러 번 실행하고 가장 좋은 결과 선택 (`n_init` 매개변수 조정).

3.2.2 밀도 기반 군집화 (DBSCAN)

개념

- 데이터가 밀집된 영역을 하나의 군집으로 정의
- 밀도가 낮은 영역은 군집 외부로 간주 (노이즈/이상치로 처리)

작동 방식

1. 각 포인트 주변 반경 `eps` 내에 `minPts` 이상의 이웃이 있으면 핵심 포인트로 간주
2. 핵심 포인트 간 연결되어 있으면 하나의 군집으로 간주
3. 밀도가 낮은 포인트는 군집 외부로 분류

장점

- 이상치에 강하고, 군집 수 K를 지정하지 않아도 됨
- 복잡한 군집 구조도 탐지 가능

단점

- `eps`, `minPts` 설정이 민감
- 고차원 데이터에는 비효율적일 수 있음

적용 팁

- **eps 탐색**: `k-distance plot`을 사용하여 급격한 변화 지점을 eps로 설정
- **minPts 기준**: 일반적으로 데이터 차원의 2배에서 4배로 설정
- **스케일링**: 거리 기반 알고리즘이므로 정규화 필수
- **군집 평가**: Silhouette Score로 군집 품질 측정 가능하나, 노이즈가 포함되므로 해석 주의

3.2.3 주성분 분석 (PCA, Principal Component Analysis)

개념

- 고차원 데이터를 저차원으로 변환해 정보를 압축
- 분산(variance)이 가장 큰 방향을 기준으로 새로운 축 생성

작동 방식

1. 데이터 정규화 (표준화)
2. 공분산 행렬 계산
3. 고유벡터/고유값을 통해 주성분 축 도출
4. 상위 N개의 축으로 데이터 투영

장점

- 데이터 시각화에 유리
- 차원 축소를 통해 연산 속도 개선, 노이즈 제거 효과

단점

- 해석이 어려움 (주성분 축의 의미 불명확)
- 선형 구조만 반영되며, 비선형 관계는 반영되지 않음

적용 팁

- **스케일링 필수**: 변수 간 단위가 다를 경우 분산 기준이 왜곡되므로 표준화 필요
- **주성분 수 선택**: 누적 설명 분산이 90~95%가 되도록 구성
- **시각화**: 2D 또는 3D 시각화로 클러스터링이나 분류 전처리로 활용 가능
- **다중 공선성 제거**: 상관관계가 높은 특성 제거 효과 → 회귀모델 성능 개선 가능