Chapter04. 분류

01. 분류의 개요

지도학습: 레이블, 즉 명시적인 정답이 있는 데이터가 주어진 상태에서 학습하는 머신러닝 방식

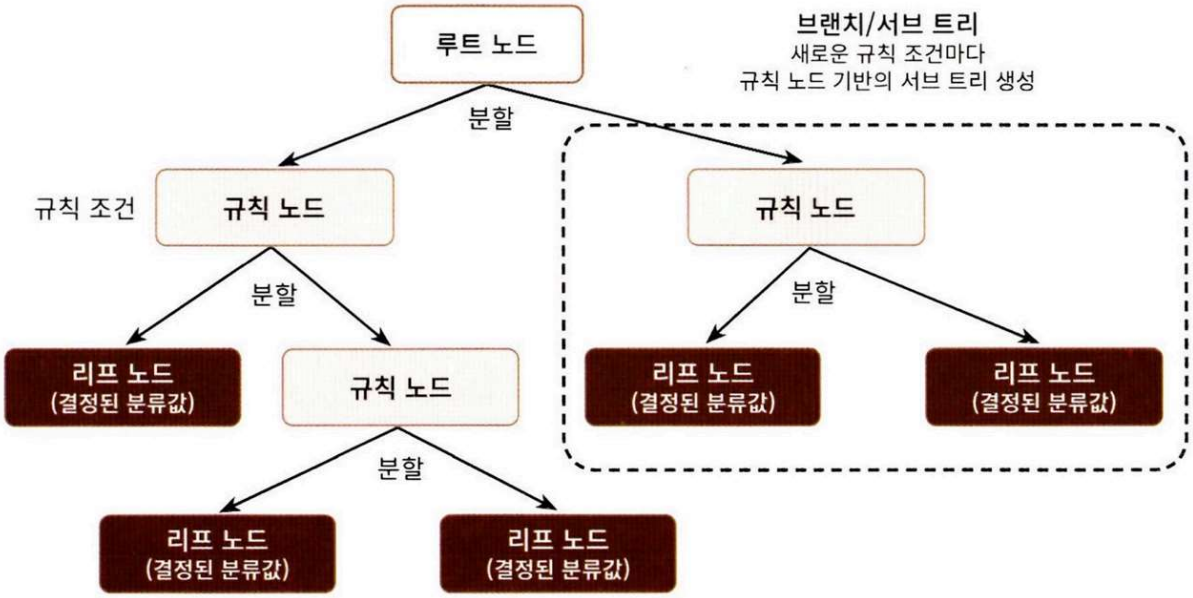
- 분류: 지도학습의 대표적 유형, 학습 데이터로 주어진 데이터의 피처와 레이블값을 머신러닝 알고리즘으로 학습해 모델을 생성하고, 새로운 데이터 값이 주어졌을 때 미지의 레이블 값을 예측
    - 베이즈 통계와 생성 모델에 기반한 나이브 베이즈
    - 독립변수와 종속변수의 선형 관계성에 기반한 로지스틱 회귀
    - 데이터 균일도에 따른 규칙 기반의 결정 트리
    - 개별 클래스 간의 최대 분류 마진을 효과적으로 찾아주는 서포트 벡터 머신
    - 근접 거리를 기준으로 하는 최소 근접 알고리즘
    - 심층 연결 기반의 신경망
    - **서로 다른 머신러닝 알고리즘을 결합한 앙상블**
        - 일반적으로 배깅과 부스팅 방식으로 나뉨
            - 배깅 방식
                - 랜덤 포레스트: 뛰어난 예측 성능, 상대적으로 빠른 수행 시간, 유연성
            - 부스팅 방식
                - 그래디언트 부스팅: 뛰어난 예측 성능, 수행 시간이 너무 오래 걸림
                - XgBoost와 LightGBM: 예측 성능 발전, 수행 시간 단축
        - 결정 트리를 기본 알고리즘으로 일반적으로 사용
            - 데이터의 스케일링이나 정규화 등의 사전 가공의 영향이 매우 적음
            - 복잡한 규칙 구조를 가져야 해서 과적합이 발생할 수도 있음

02. 결정 트리

결정 트리: 데이터에 있는 규칙을 학습을 통해 자동으로 찾아내 트리 기반의 분류 규칙을 만드는 것

- 일반적으로 if/else 기반으로 규칙을 가장 쉽게 표현
- 결정 트리의 구조
    - 규칙 노드: 규칙 조건
    - 리프 노드: 결정된 클래스 값

    - 서브 트리: 새로운 규칙 조건마다 생성
    - 데이터 세트에 피처가 있고, 피처가 결합해 규칙 조건을 만들 때마다 규칙 노드가 만들어짐
    - 트리의 깊이가 깊어질수록 결정 트리의 예측 성능이 저하될 가능성이 높음
    - 최대한 균일한 데이터 세트를 구성할 수 있도록 분할하는 것이 필요



- 결정 노드는 정보 균일도가 높은 데이터 세트를 먼저 선택할 수 있도록 규칙 조건을 만듦
- 정보 균일도가 데이터 세트로 쪼개질 수 있도록 조건을 찾아 서브 데이터 세트를 만들고, 이 서브 데이터 세트에서 균일도가 높은 자식 데이터 세트로 쪼개는 방식으로 자식 트리로 내려가면서 반복하는 방식으로 데이터 값을 예측
- 정보 균일도를 측정하는 방법: 엔트로피를 이용한 정보 이득 지수, 지니 계수
    - 정보 이득 지수: 1에서 엔트로피 지수를 뺀 값, 결정 트리는 정보 이득이 높은 속성을 기준으로 분할
    - 지니 계수: 낮을수록 데이터 균일도가 높은 것으로 해석, 지니 계수가 낮은 속성을 기준으로 분할
- 데이터가 모두 특정 분류에 속하게 되면 분할을 멈추고 분류를 결정

- 결정 트리 모델의 특징
    - 정보의 '균일도'라는 룰을 기반으로 하고 있어서 알고리즘이 쉽고 직관적
    - 어떻게 규칙 노드와 리프 노드가 만들어지는지 알 수 있고, 시각화로 표현 가능
    - 각 피처의 스케일링과 정규화 같은 전처리 작업 필요 없음
    - 과적합으로 정확도가 떨어진다는 단점, 트리의 크기를 사전에 제한하는 튜닝 필요

- 결정 트리 파라미터
    - min_samples_split

        - 노드를 분할하기 위한 최소한의 샘플 데이터 수로 과적합을 제어하는 데 사용
        - 디폴트는 2이고 작게 설정할수록 분할되는 노드가 많아져서 과적합 가능성 증가
        - 과적합을 제어, 1로 설정할 경우 분할되는 노드가 많아져서 과적합 가능성 증가
    - min_samples_leaf
        - 말단 노드가 되기 위한 최소한의 샘플 데이터 수
        - min_samples_split와 유사하게 과적합 제어 용도, 그러나 비대칭적 데이터의 경우 특정 클래스의 데이터가 극도로 작을 수 있으므로 이 경우는 작게 설정 필요
    - max_features
        - 최적의 분할을 위해 고려할 최대 피처 개수
        - int형으로 지정하면 대상 피처의 개수, float형을 지정하면 전체 피처 중 대상 피처의 퍼센트
    - max_depth
        - 트리의 최대 깊이를 규정
        - 깊이가 깊어지면 min_samples_split 설정대로 최대 분할하여 과적합할 수 있으므로 적절한 값으로 제어 필요
    - max_leaf_nodes
        - 말단 노드의 최대 개수

- 결정 트리 모델의 시각화
    - Graphviz 패키지: 원래 그래프 기반의 dot 파일로 기술된 다양한 이미지를 쉽게 시각화할 수 있는 패키지
    - Graphviz 패키지와 쉽게 인터페이스 할 수 있도록 export_graphviz( ) API 제공
        - 함수 인자로 학습이 완료된 Estimator, 피처의 이름 리스트, 레이블 이름 리스트를 입력하면 학습된 결정 트리 규칙을 실제 트리 형태로 시각화해 보여줌
    - 리프 노드: 더 이상 자식 노드가 없는 노드
        - 최종 클래스 값이 결정
        - 오직 하나의 클래스 값으로 최종 데이터가 구성되거나 리프 노드가 될 수 있는 하이퍼 파라미터 조건을 충족하면 됨
    - 브랜치 노드: 자식 노드가 있는 노드
    - 규칙 생성 로직을 미리 제어하지 않으면 완벽하게 클래스 값을 구별해내기 위해 트리 노드를 계속해서 만듦, 과적합 되어버림
    - 결정 트리 알고리즘이 학습을 통해 규칙을 정하는 데 있어 피처의 중요한 역할 지표를 DecisionTreeClassifier 객체의 feature_importances_ 속성으로 제공
        - ndarray 형태로 값을 반환하며 피처 순서대로 값이 할당됨, 값이 높을수록 해당 피처의 중요도가 높다는 의미

- 결정 트리 과적합
    - 분류를 위한 테스트용 데이터를 쉽게 만들 수 있도록 make_classification( )함수를 제공
        - 호출 시 반환되는 객체는 피처 데이터 세트와 클래스 레이블 데이터 세트
    - 결정 트리의 기본 하이퍼 파라미터 설정은 리프 노드 안에 데이터가 모두 균일하거나 하나만 존재해야 하는 엄격한 분할 기준으로 인해 결정 기준 경계가 많아지고 복잡해짐
    - 테스트 데이터 세트는 학습 데이터 세트와는 다른 데이터 세트인데, 학습 데이터에만 지나치게 최적화된 분류 기준은 정확도를 떨어뜨릴 수 있음

03. 앙상블 학습

앙상블 학습을 통한 분류: 여러 개의 분류기를 생성하고 그 예측을 결합함으로써 보다 정확한 최종 예측을 도출하는 기법

- 앙상블 알고리즘의 대표격인 랜덤 포레스트와 그래디언트 부스팅 알고리즘은 뛰어난 성능과 쉬운 사용, 다양한 활용도로 인해 많이 애용
- 부스팅 계열의 인기와 강세가 계속 이어져 기존의 그래디언트 부스팅을 뛰어넘는 새로운 알고리즘의 개발이 가속화
    - XGboost: 오픈 플랫폼인 캐글에서 '매력적인 솔루션'으로 불림
    - LightGBM: XGboost와 유사한 예측 성능을 가지면서도 훨씬 빠른 수행 속도를 가짐
    - 스태킹: 여러 가지 모델의 결과를 기반으로 메타 모델을 수립
- 앙상블 학습의 유형: 보팅, 배깅, 부스팅 등
    - 보팅과 배깅: 여러 개의 분류기가 투표를 통해 최종 예측 결과를 결정하는 방식
        - 보팅: 일반적으로 서로 다른 알고리즘을 가진 분류기를 결합
        - 배깅: 각각의 분류기가 모두 같은 유형의 알고리즘, 데이터 샘플링을 서로 다르게 가져가면서 학습을 수행해 보팅을 수행
            - 부트스트래핑 분할 방식: 개별 Classifier에게 데이터를 샘플링해서 추출하는 방식
            - 배깅 앙상블 방식: 개별 분류기가 부트스트래핑 방식으로 샘플링된 데이터 세트에 대해서 학습을 통해 개별적인 예측을 수행한 결과를 보팅을 통해서 최종 예측 결과를 선정하는 방식
            - 배깅 방식은 중첩을 허용
        - 부스팅: 여러 개의 분류기가 순차적으로 학습을 수행하되, 앞에서 학습한 분류기가 예측이 틀린 데이터에 대해서는 올바르게 예측할 수 있도록 다음 분류기에게는 가중치를 부여하면서 학습과 예측을 진행
            - 그래디언트 부스트, XGBoost, LightBGM

        - 스태킹: 여러 가지 다른 모델의 예측 결괏값을 다시 학습 데이터로 만들어서 다른 모델로 재학습시켜 결과를 예측하는 방법

- 보팅 유형
    - 하드 보팅: 예측한 결괏값들 중 다수의 분류기가 결정한 예측값을 최종 보팅 결괏값으로 선정
    - 소프트 보팅: 분류기들의 레이블 값 결정 확률을 모두 더하고 이를 평균해서 이들 중 확률이 가장 높은 레이블 값을 최종 보팅 결괏값으로 선정
- 보팅 분류기
    - 사이킷런은 보팅 방식의 앙상블을 구현한 VotingClassifier 클래스를 제공
        - 이 클래스를 이용해 보팅 분류기 생성 가능
    - 보팅으로 여러 개의 기반 분류기를 결합한다고 해서 무조건 기반 분류기보다 예측 성능이 향상되는 것 아님, 데이터 특성과 분포 등 다양한 요건에 따라 오히려 기반 분류기 중 가장 좋은 분류기의 성능이 보팅했을 때보다 나을수도 있음

04. 랜덤 포레스트

- 배깅: 같은 알고리즘으로 여러 개의 분류기를 만들어서 보팅으로 최종 결정하는 알고리즘, 대표적으로 랜덤 포레스트
- 랜덤 포레스트: 여러 개의 결정 트리 분류기가 전체 데이터에서 배깅 방식으로 각자의 데이터를 샘플링해 개별적으로 학습을 수행한 뒤 최종적으로 모든 분류기가 보팅을 통해 예측 결정
    - 개별적인 분류기의 기반 알고리즘은 결정 트리이지만 개별 트리가 학습하는 데이터 세트는 전체 데이터에서 일부가 중첩되게 샘플링된 데이터 세트
    - 랜덤 포레스트의 서브세트 데이터는 부트스트래핑으로 데이터가 임의로 만들어짐
    - RandomForestClassifier 클래스를 통해 랜덤 포레스트 기반의 분류를 지원

- 랜덤 포레스트 하이퍼 파라미터 및 튜닝
    - n_estimators: 랜덤 포레스트에서 결정 트리의 개수를 지정
    - max_features: 결정 트리에 사용된 max_features 파라미터와 같으나 기본 max_features는 'None'이 아니라 'auto'임
    - max_depth, min_samples_leaf: 과적합을 개선하기 위해 사용