Euron_ML_Week4

생성일	@October 6, 2024 3:02 AM

4. 분류

01. 분류(Classification)의 개요

💡 지도학습의 대표적인 유형

> 학습 데이터로 주어진 데이터의 피처와 레이블값을 머신러닝 알고리즘으로 학습해 모델 생성
>
> 생성된 모델의 새로운 데이터 값이 주어졌을 때 미지의 레이블 값 예측
>
> ⇒ 기존 데이터가 어떤 어떤 레이블에 속하는지 패턴을 알고리즘으로 인지한 뒤에 새롭게 관측된 데이터에 대한 레이블을 판별하는 것

[앙상블]

— 배깅(Bagging)

: 랜덤 포레스트(Random Forest)

— 부스팅(Boosting)

: 그래디언트 부스팅(Gradient Boosting)

---

02. 결정 트리

💡 데이터에 있는 규칙을 학습을 통해 찾아내 트리 기반의 규칙을 만드는 것

: if-else를 자동으로 찾아내 예측을 위한 규칙을 만드는 알고리즘

→ 데이터의 어떤 기준을 바탕으로 규칙을 만들어야 가장 효율적인 분류가 될 것인가!

[구조]

- 규칙 노드(Decision Node)

  : 규칙 조건이 됨
- 리프 노드(Leaf Node)

  : 결정된 클래스 값
- 브랜치/서브트리

  : 새로운 규칙 조건마다 규칙 노드 기반의 서브 트리 생성

  : (많은 규칙→ 분류 결정 방식 복잡→ 과적합) 트리의 깊이가 깊어질수록 결정 트리의 예측 성능이 저하될 가능성 높음

[분할]

> 최대한 균일한 데이터 세트를 구성할 수 있도록 (혼잡도<균일도)
>
> 결정 노드: 정보 균일도가 높은 데이터 세트를 먼저 선택할 수 있도록 규칙 조건을 만듦 → 균일도가 높은 자식 데이터 세트 쪼개는 방식을 자식 트리로 내려가면서 반복하는 방식

- 정보 균일도 측정 방식
  - 엔트로피를 이용한 정보 이득(Information Gain) 지수

    : 엔트로피는 주어진 데이터 집합의 혼잡도 의미

    (서로 다른 값이 섞여 있으면 엔트로피가 높고, 같은 값이 섞여 있으면 엔트로피가 낮음)

    : 정보 이득 지수= 1- 엔트로피 지수

    : 결정 트리는 정보 이득이 높은 속성을 기준으로 분할
  - 지니계수

    : (유래_ 경제학) 불평등 지수를 나타낼 때 사용하는 계수

    : 0이 가장 평등하고 1로 갈수록 불평등

    : 지니 계수가 낮을수록 데이터 균일도가 높다

    : 지니 계수가 낮은 속성을 기준으로 분할

- 정보 이득이 높거나 지니 계수가 낮은 조건을 찾아 자식 트리 노드에 걸쳐 반복적으로 분할→ 데이터가 모두 특정 분류에 속하게 되면 분할을 멈추고 분류를 결정

- 결정 트리 모델의 특징

[장점]

- '균일도'라는 룰→ 알고리즘이 쉽고 직관적임
- 결정 트리의 룰이 매우 명확→ 규칙/리프 노드가 만들어지는지 알 수 있고 시각화로 표현 가능
- 정보의 균일도만 신경쓰면 됨→ 피처의 스케일링이나 정규화 등의 사전 가공 영향도가 크지 않음

[단점]

- '균일도'라는 룰→ 피처가 많고 균일도가 다양하게 존재

  → 과적합으로 정확도가 떨어짐

  ⇒ 이를 극복하기 위해 트리의 크기를 사전에 제한하는 튜닝 필요

- 결정 트리 파라미터

[클래스]

분류: DecisionTreeClassifier

회귀: DecisionTreeRegressor

[파라미터]

- min_samples_split
  - 노드를 분할하기 위한 최소한의 샘플 데이터 수로 과적합을 제어하는 데 사용
  - 디폴트는 2, 작게 설정할수록 분할되는 노드가 많아져 과적합 가능성 증가

- min_samples_leaf
  - 말단 노드(Leaf)가 되기 위한 최소한의 샘플 데이터 수

    - 과적합 제어 용도(min_samples_split과 유사)
    - 비대칭적 데이터의 경우) 클래스의 데이터가 극도로 작을 수 있으므로 작게 설정 필요

- max_features
    - 최적의 분할을 위해 고려할 최대 피처 개수
    - 디폴트는 None, 데이터 세트의 모든 피처를 사용해 분할 수행
        - int형 지정→ 대상 피처의 개수
        - float형 지정→ 전체 피처 중 대상 피처의 퍼센트
        - ‘sqrt’→ (전체 피처 개수)**(1/2)
        - ‘auto’→ sqrt와 동일
        - ‘log’→ 전체 피처 중 log2(전체 피처 개수) 선정
        - ‘None’→ 전체 피처 선정

- max_depth
    - 트리의 최대 깊이 규정
    - 디폴트는 None

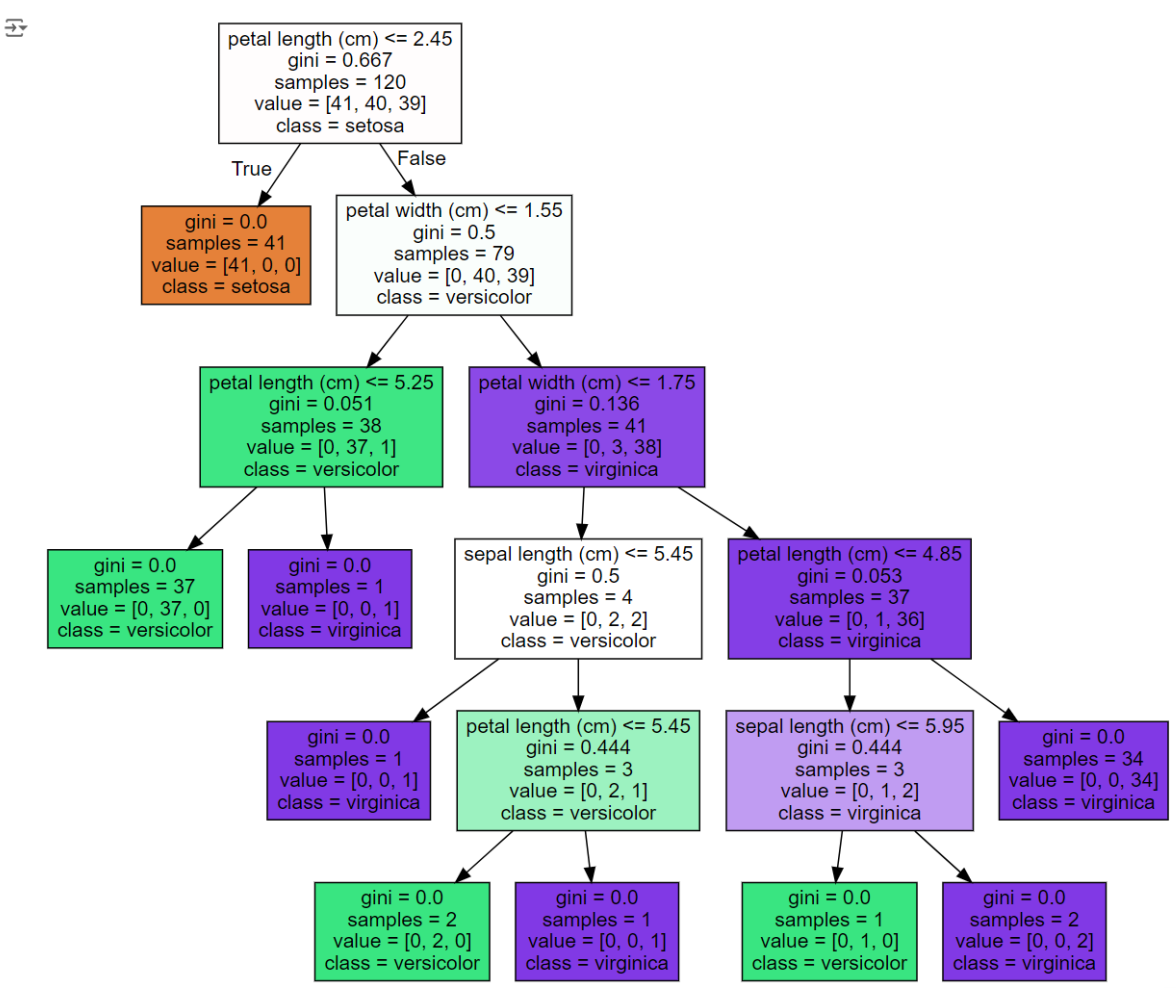
      → 완벽하게 클래스 결정 값이 될 때까지 깊이를 계속 키우며 분할

      → 노드가 가지는 데이터 개수가 min_samples_split보다 작아질 때까지 계속 깊이를 증가시킴
    - 깊이가 깊어지면 min_samples_split 설정대로 최대 분할하여 과적합할 수 있으므로 적절한 값으로 제어 필요

- max_leaf_nodes
    - 말단 노드(Leaf)의 최대 개수

- 결정 트리 모델의 시각화

> Graphviz 패키지 사용



▼ 지표의 의미

- petal length(cm) <=2.45와 같이 피처의 조건이 있는 것

  : 자식 노드를 만들기 위한 규칙 조건

  : 조건이 없으면 리프 노드

- gini

  : 다음의 value=[ ]로 주어진 데이터 분포에서의 지니 계수

- samples

  : 현 규칙에 해당하는 데이터 건수

- value=[ ]

  : 클래스 값 기반의 데이터 건수

  : [0, 1, 2]→ 0: Setosa, 1: Verdicolor, 2: Virginica

  [41, 40, 39]→ 41: Setosa, 40: Verdicolor, 39: Virginica

  개로 데이터가 구성돼 있다는 의미

▼ 각 노드의 의미

- 2번 노드
  - petal length (cm) <= 2.45 규칙이 True로 분기

    : 41개의 샘플 데이터 모두 Setosa로 결정되므로 클래스가 결정된(예측 클래스는 Setosa로 결정) 리프 노드→ 더 이상 규칙을 만들 필요가 없음

    : 지니 계수는 0

- 3번 노드
  - petal length (cm) <= 2.45 규칙이 False로 분기

    : 79개의 샘플 데이터 중 Versicolor 40개, Virginica 39개

    : 지니 계수는 0.5→ 높으므로 다음 자식 브랜치 노드로 분기할 규칙 필요

    → petal length (cm) <= 1.55 규칙으로 자식 노드 생성

- 4번 노드

  : 38개의 샘플 데이터 중 Versicolor 37개, Virginica 1개→ 대부분 versicolor

  : 지니 계수는 0.051→ 매우 낮으나 여전히 혼재되어 있음

  → petal length (cm) <= 5.25 규칙으로 자식 노드 생성

- 5번 노드

  : 41개의 샘플 데이터 중 Versicolor 3개, Virginica 38개→ 대부분 virginica

  : 지니 계수는 0.136→ 낮으나 여전히 혼재되어 있음

  → petal length (cm) <= 1.75 규칙으로 자식 노드 생성

▼ 하이퍼 파라미터 변경 시

- max_depth하이퍼 파라미터 변경

  : 결정 트리의 최대 트리 깊이 제어→ 더 간단한 결정 트리 생성

- min_samples_split 하이퍼 파라미터 변경

  : 자식 규칙 노드를 분할해 만들기 위한 최소한의 샘플 데이터 개수→ 트리 깊이가 줄고 더 간결한 결정 트리 생성

- min_samples_leaf 하이퍼 파라미터 변경

  : 리프 노드가 될 수 있는 샘플 데이터 건수의 최솟값 지정→ 자연스럽게 브랜치 노드가 줄고 결정 트리가 더 간결하게 만들어짐

- feature_importances_

: 사이킷런- 결정 트리 알고리즘이 학습을 통해 규칙을 정할 수 있도록 피처의 중요한 역할 지표를 속성으로 제공

: 값이 높을수록 해당 피처의 중요도가 높음

: 피처별로 결정 트리 알고리즘에서 중요도를 추출할 수 있게 함

- 결정 트리 과적합(Overfitting)

: colab에서 실행

- 결정 트리 실습- 사용자 행동 인식 데이터 세트

: colab에서 실행

03. 앙상블 학습

- 앙상블 학습 개요

> 앙상블 학습을 통한 분류
>
> : 여러 개의 분류기를 생성하고 그 예측을 결합함으로써 보다 정확한 최종 예측을 도출하는 기법
>
> ⇒ 예측 결과를 결합함으로써 단일 분류기보다 신뢰성이 높은 예측값을 얻음

[앙상블 학습의 유형]

- 여러 개의 분류기가 투표를 통해 최종 예측 결과를 결정
  - 보팅(Voting)

    : 일반적으로 서로 다른 알고리즘을 가진 분류기 결합

    : Linear Regression, K Nearest Neighbor, Support Vector Machine
  - 배깅(Bagging)

    : 각각의 분류기가 모두 같은 유형의 알고리즘 기반

    : 데이터 샘플링을 서로 다르게 가져가면서 학습을 수행해 보팅을 수행

    : 부스트래핑 분할 방식: 개별 Classifier에게 데이터를 샘플링해서 추출

    : 대표적- *랜덤 포레스트 알고리즘*
- 여러 개의 분류기가 순차적으로 학습을 수행하되, 앞에서 학습한 분류기가 예측이 틀린 데이터에 대해서는 올바르게 예측할 수 있도록 가중치를 부여하면서 학습과 예측을 진행
  - 부스팅(Boosting)
  - 대표적- *그래디언트 부스트, XGBoost, LightGBM*

- 보팅 유형- 하드 보팅(Hard Voting)과 소프트 보팅(Soft Voting)

- 하드 보팅을 이용한 분류

  : 예측한 결괏값들 중 다수의 분류기가 결정한 예측값을 최종 보팅 결괏값으로 선정
- 소프트 보팅을 이용한 분류

  : 분류기들의 레이블 값 결정 확률을 모두 더하고 이를 평균해서 이들 중 확률이 가장 높은 레이블 값을 최종 보팅 결괏값으로 선정

- 보팅 분류기(Voting Classifier)

: colab에서 실행

04. 랜덤 포레스트

- 랜덤 포레스트의 개요 및 실습

> 💡 배깅의 대표적인 알고리즘

: 여러 개의 분류기를 만들어 보팅으로 최종 결정

: 비교적 빠른 수행 속도, 다양한 영역에서 높은 예측 성능

1. 여러 개의 결정 트리 분류기가 전체 데이터에서 배깅 방식으로 각자의 데이터를 샘플링
2. 개별적으로 학습을 수행
3. 모든 분류기가 보팅을 통해 예측 결정

- 부트스트래핑(bootstrapping) 분할 방식

  : 여러 개의 데이터 세트를 중첩되게 분리하는 것

  : 랜덤 포레스트는 개별 트리가 학습하는 데이터 세트가 전체 데이터에서 일부가 중첩되게 샘플링된 데이터 세트임

  → subset데이터는 부트스트래핑으로 임의적으로 만들어진다

- RandomForestClassifier 클래스

  : 사이킷런은 위 클래스를 통해 랜덤 포레스트 기반의 분류를 지원

- 랜덤 포레스트 하이퍼 파라미터 및 튜닝

[하이퍼 파라미터]

- 결정 트리에서 사용되는 하이퍼 파라미터와 유사
  - n_estimators

    : 랜덤 포레스트에서 결정 트리의 개수 지정

    : 디폴트는 10개

    : 많이 설정할수록 좋은 성능 기대 가능 but 계속 증가한다고 무조건 성능이 향상되는 것은 아님

    : 늘릴수록 학습 수행 시간이 오래 걸림
  - max_features

    : 결정 트리에 사용된 max_features 파라미터와 같음

    : **차이점) '**None'이 아니라 'auto' 즉, 'sqrt'와 같음

    → 랜덤 포레스트의 트리를 분할하는 피처를 참조할 때 전체 피처가 아니라 sqrt(전체 피처 개수)만큼 참조

    ex) 전체 피처가 16개라면 분할을 위해 4개 참조
  - max_depth, min_samples_leaf

    : 결정 트리에서 과적합을 개선하기 위해 사용되는 파라미터가 랜덤 포레스트에서도 똑같이 적용될 수 있음

- GridSearchCV를 이용한 파라미터 튜닝

  : colab에서 실행