

파머완 4장 Part 1 필사 & 개념정리

4-1 분류의 개요

분류는? 지도학습의 대표적 유형으로 학습 데이터로 주어진 데이터의 피처와 레이블값을 머신러닝 알고리즘으로 학습→모델 생성→새로운 데이터값을 이용해 미지의 레이블 값을 '예측'

분류의 머신러닝 알고리즘:

- 베이즈(Bayes) 통계와 생성 모델에 기반한 나이브 베이즈(Naive Bayes)
- 독립변수와 종속변수의 선형 관계성에 기반한 로지스틱 회귀(Logistic Regression)
- 데이터 균일도에 따른 규칙 기반의 결정 트리(Decision Tree)
- 개별 클래스 간의 최대 분류 마진을 효과적으로 찾아주는 서포트 벡터 머신(Support Vector Machine)
- 근접 거리를 기준으로 하는 최소 근접(Nearest Neighbor) 알고리즘
- 심층 연결 기반의 신경망(Neural Network)
- 서로 다른(또는 같은) 머신러닝 알고리즘을 결합한 양상블(Ensemble) → 4장에서 다룸!

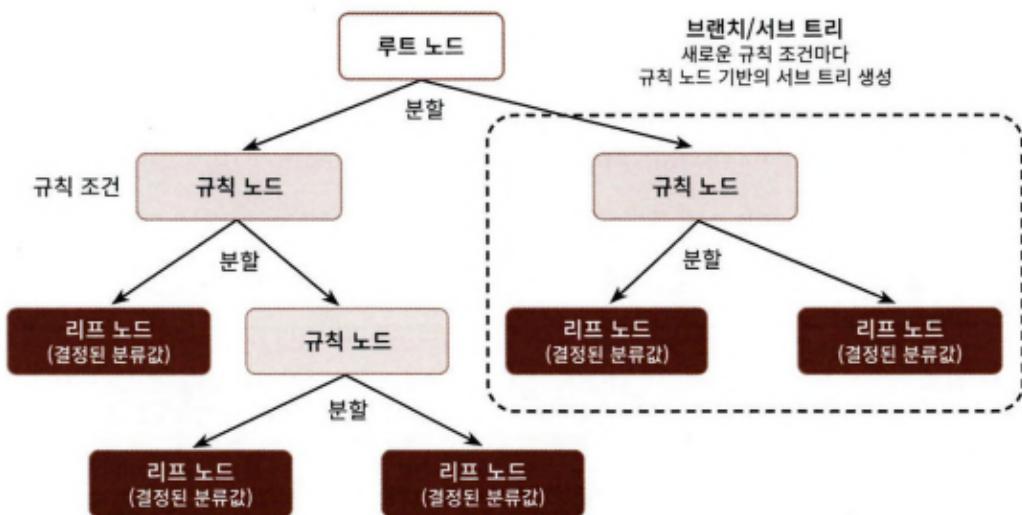
양상블: 정형 데이터의 예측 분석 영역에서는 양상블이 매우 높은 예측 성능으로 인해 많은 분석가와 데이터 과학자들에게 애용

- **Bagging:** 대표적인 예→랜덤 포레스트(Random Forest)는 뛰어난 예측 성능, 상대적으로 빠른 수행 시간, 유연성
- **Boosting:** 그래디언트 부스팅(Gradient Boosting)의 경우 뛰어난 예측 성능을 가지 고 있지만, 수행 시간이 너무 오래 걸리는 단점으로 인해 최적화 모델 튜닝이 어려웠음. XgBoost(eXtra Gradient Boost)와 LightGBM 등 기존 그래디언트 부스팅의 예측 성능을 한 단계 발전시키면서도 수행 시간을 단축시킨 알고리즘이 계속 등장하면서 정형 데이터의 분류 영역에서 가장 활용도가 높은 알고리즘으로 자리 잡았습니다.
- 양상블은 서로 다른 혹은 같은 알고리즘을 결합하는데 보통 같은 알고리즘을 결합하고, 결정 트리로 결합함.
- 왜? 양상블은 매우 많은 여러 개의 약한 학습기를 결합해 확률적 보완, 가중치 계속 업데이트해서 예측 성능을 향상하는데, 결정트리는 바로 이 약한 학습기의 역할을 잘 수행함.

4-2 결정 트리

마치 if/else 처럼 데이터에 있는 규칙을 학습을 통해 자동으로 찾아내 트리 기반의 분류 규칙을 만드는 것.

구성: 규칙노드와 리프 노드, 서브 트리



이렇게 데이터 세트에 피처가 있고, 피처를 결합해 규칙 조건을 만들 때마다 규칙 노드가만 들어짐. ⇒ 규칙 노드가 많아지며 깊이가 길어지고, 너무 복잡해지면서 과적합으로 이어지기 쉬움.

⇒ 어떻게 트리를 **분할(Split)**할 것인가가 중요한데 최대한 균일한 데이터 세트를 구성할 수 있도록 분할하는 것이 필요

결정노드: 정보 균일도가 높은 데이터 세트를 먼저 선택할 수 있게 규칙 조건을 만들어줌. 정보 균일도가 높은 순서로 세트를 쪼개서 반복하는 방식으로 데이터 값 예측.

정보 균일도를 측정하는 방법:

1. 엔트로피를 이용한 정보 이득 지수

서로 다른 값이 섞여 있으면 엔트로피가 높

같은 값이 섞여 있으면 엔트로피가 낮

정보 이득 지수는 1에서 엔트로피 지수를 뺀 값입니다. 즉. $1 - \text{엔트로피 지수}$ 입니다. 결정 트리는 이 정보 이득 지수로 분할 기준을 정합니다. 즉, 정보 이득이 높은 속성을 기준으로 분할

2. 지니 계수

데이터 균일도 높 → 지니 계수 낮

0이 가장 평등하고 1로 갈수록 불평등합니다. 지니 계수가 낮을수록 데이터 균일도가 높은 것으로 해석해 지니 계수가 낮은 속성을 기준으로 분할합니다.

⇒ DecisionTreeClassifier는 기본으로 지니 계수를 이용

정보 이득이 높거나 지니 계수가 낮은 조건을 찾아서 자식 트리 노드에 걸쳐 반복적으로 분할한

뒤, 데이터가 모두 특정 분류에 속하게 되면 분할을 멈추고 분류를 결정합니다.

결정 트리 모델의 특징:

장점:

1.'균일도'라는 룰을 기반이라 알고리즘이 쉽고 직관.

2.피처의 스케일링과 정규화 필요 없음

단점:

과적합으로 정확도가 떨어짐. → 서브 트리를 계속 만들다 보면 피처가 많고 균일도가 다양하게 존재할수록 트리의 깊이가 커지고 복잡 → 트리의 크기를 사전에 제한하는 것이 오히려 성능 튜닝에 필요

결정 트리 파라미터:

사이킷런의 결정 트리 알고리즘 구현 클래스: DecisionTreeClassifier(분류), DecisionTreeRegressor(회귀)

위 두 클래스는 다음과 같은 파라미터를 사용함.

파라미터 명	설명
min_samples_split	<ul style="list-style-type: none">노드를 분할하기 위한 최소한의 샘플 데이터 수로 과적합을 제어하는 데 사용됨.디폴트는 2이고 작게 설정할수록 분할되는 노드가 많아져서 과적합 가능성 증가
min_samples_leaf	<ul style="list-style-type: none">분할이 될 경우 왼쪽과 오른쪽의 브랜치 노드에서 가져야 할 최소한의 샘플 데이터 수큰 값으로 설정될수록, 분할될 경우 왼쪽과 오른쪽의 브랜치 노드에서 가져야 할 최소한의 샘플 데이터 수 조건을 만족시키기가 어려우므로 노드 분할을 상대적으로 멀 수함.min_samples_split와 유사하게 과적합 제어 용도. 그러나 비대칭적(imbalanced) 데이터의 경우 특정 클래스의 데이터가 극도로 작을 수 있으므로 이 경우는 작게 설정 필요.

파라미터 명	설명
max_features	<ul style="list-style-type: none"> 최적의 분할을 위해 고려할 최대 피처 개수. 디폴트는 None으로 데이터 세트의 모든 피처를 사용해 분할 수행. int 형으로 지정하면 대상 피처의 개수, float 형으로 지정하면 전체 피처 중 대상 피처의 퍼센트임 'sqrt'는 전체 피처 중 $\sqrt{\text{전체 피처 개수}} \times \text{전체 피처 개수 만큼}$ 선정 'auto'로 지정하면 sqrt와 동일 'log'는 전체 피처 중 $\log_2(\text{전체 피처 개수})$ 선정 'None'은 전체 피처 선정
max_depth	<ul style="list-style-type: none"> 트리의 최대 깊이를 규정. 디폴트는 None. None으로 설정하면 완벽하게 클래스 결정 값이 될 때까지 깊이를 계속 키우며 분할하거나 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	<ul style="list-style-type: none"> 말단 노드(Leaf)의 최대 개수

결정 트리 모델의 시각화

어떤 규칙을 가지고 트리를 생성하는지를 시각적으로 보면 좋겠지? Graphviz 패키지를 다운 받아 사용하면 됨.

시각화된 트리에서는 각 노드의 규칙 조건, 지니 계수, 샘플 데이터 건수, 그리고 결정된 클래스 값 등을 확인할 수 있음.

결정 트리 알고리즘은 `feature_importances_` 속성을 통해 피처의 중요도를 제공합니다. 이 값은 피처가 트리 분할 시 정보 이득이나 지니 계수를 얼마나 효율적으로 개선했는지를 정규화된 값으로 표현하며, 값이 높을수록 중요도가 높습니다

Graphviz 이용하기

리프 노드: 더 이상 자식이 없는 노드 → 최종 클래스 값이 결정됨.

브랜치 노드: 자식이 있는 노드

- `petal length(cm) <= 2.45`와 같이 피처의 조건이 있는 것은 자식 노드를 만들기 위한 규칙 조건입니다. 이 조건이 없으면 리프 노드입니다.
- `gini`는 다음의 `valued`]로 주어진 데이터 분포에서의 지니 계수입니다.
- `samples`는 현 규칙에 해당하는 데이터 건수입니다.

- `value = []`는 클래스 값 기반의 데이터 건수입니다. 붓꽃 데이터 세트는 클래스 값으로 0, 1, 2를 가지고 있으며, 0 : Setosa, 1 : Versicolor, 2 : Virginica 품종을 가리킵니다. 만일 `Value = [41, 40, 39]`라면 클래스 값의 순서로 Setosa 41개, Versicolor 40 개, Virginica 39개로 데이터가 구성돼 있다는 의미입니다.
- 각 노드의 색깔은 붓꽃 데이터의 레이블 값을 의미함. 색깔이 짙어질수록 지니 계수가 낮고 해당 레이블에 속하는 샘플 데이터가 많다는 의미
- `max_depth` : 트리의 최대 깊이 제한. 값이 작을수록 단순한 트리.
- `min_samples_split` : 자식 노드를 만들기 위한 최소 샘플 수. 예: 값이 4일 때 노드에 샘플이 3개면 분할하지 않고 리프 노드로 종료
- `min_samples_leaf` : 리프 노드가 되기 위해 가져야 하는 최소 샘플 수. 값이 커질수록 리프 노드 생성이 어려워지고, 트리가 간결해짐.
- 사이킷런의 `DecisionTreeClassifier` 는 학습 후 `feature_importances_` 속성으로 **피처 중요도**를 제공
- 중요도 값은 **트리 분할 시 지니 계수/정보 이득 개선 효과를 정규화한 값**

결정 트리 과적합

`make_classification()` → 2개의 피처가 3가지 유형의 클래스 값을 가지는 데이터 세트를 만들고 이를 그래프 형태로 시각화할 수 있게 해줍니다.

1. 결정 트리의 하이퍼 파라미터를 디폴트로 한 뒤, 결정 트리 모델이 어떠한 결정 기준을 가지고 분할하면서 데이터를 분류하는지 확인

`visualize_boundary()` 머신러닝 모델이 클래스 값을 예측하는 결정 기준을 색상과 경계로 나타내 모델이 어떻게 데이터 세트를 예측 분류하는지 잘 이해할 수 있게 해줍니다.

→ 결과 : 결정 기준 경계가 많아지고 복잡해졌습니다. 이렇게 복잡한 모델은 학습 데이터 세트의 특성과 약간만 다른 형태의 데이터 세트를 예측하면 예측 정확도가 떨어짐.

2. `min_samples_leaf = 6`으로 트리 생성 조건 제공.

이상치에 크게 반응하지 않으면서 좀 더 일반화된 분류 규칙에 따라 분류 → 1번보다 훨씬 성능이 좋다고 할 수 있음.

결정 트리 실습 - 사용자 행동 인식 데이터 세트

UCI 머신러닝 리포지토리(Machine Learning Repository)에서 제공하는 사용자 행동 인식(Human Activity Recognition) 데이터 세트에 대한 예측 분류를 수행

4-3 양상블 학습

여러 개의 분류기를 생성하고 그 예측을 결합함으로써 보다 정확한 최종 예측을 도출하는 기법.

양상블: 대부분의 정형 데이터 분류에서 사용.

양상블의 학습 유형:

1. **voting**: 여러 개의 분류기가 투표를 통해 최종 예측 결과를 결정, 일반적으로 서로 다른 알고리즘을 가진 분류기를 결합.

2. **bagging**: 분류기가 투표를 통해 최종 예측 결과를 결정하는 것은 동일하나 각각의 분류기가 모두 같은 유형의 알고리즘 기반이나, 데이터 샘플링을 서로 다르게 가져가면서 학습을 수행해 보팅 수행 → 대표적인 방식이 **랜덤 포레스트 알고리즘**.

개별 Classifier에게 데이터를 샘플링해서 추출하는 방식을 부트스트래핑(Bootstrapping) 분할 방식 이용교차 검증이 이라고 하는데 이를 이용. 이라고 하는데 이를 ○

교차 검증과 달리 중첩을 허용함.

3. **boosting**: 부스팅은 여러 개의 분류기가 순차적으로 학습을 수행하되, 앞에서 학습한 분류기가 예측이 틀린 데이터에 대해서는 올바르게 예측할 수 있게 다음 분류기에게는 가중치를 부여해서 학습과 예측을 진행. → 대표적인 모듈 **그래디언트 부스트**, XGBoost(eXtra Gradient Boost), LightGBM(Light Gradient Boost)

4. **Stacking**: 여러 가지 다른 모델의 예측 결괏값을 다시 학습 데이터로 만들어 다른 모델(메타 모델)로 재학습시켜 결과를 예측하는 방법입니다

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

1. 하드 보팅: 다수결

2. 소프트 보팅: 레이블 값 결정 확률을 모두 더하고 이를 평균. → 확률이 가장 높은 레이블 값을 최종 보팅 결괏값으로 일반적으로 소프트 보팅 채택

보팅 분류기 (사이킷런 VotingClassifier 클래스)

-보팅 분류기가 정확도가 조금 높게 나타났는데, 보팅으로 여러 개의 분류기를 결합한다고 조건 기반 분류기보다 예측 성능이 향상되지는 않음. 데이터 특성과 분포에 따라 다를 수 있음

-그러나 양상블 방법은 전반적으로 뛰어난 예측 성능을 보여줌. 유연성이 높기 때문임.

-보팅과 스태킹은 서로 다른 알고리즘이지만 배깅과 부스팅은 결정트리니까 과적합 논란이 있을 수 있음. 그러나 이것을 수십 수천개의 분류기가 결합해 지면 또 다양성이 보장돼서 예측 성능이 높아짐.

4-4 랜덤 포레스트 (bagging)

같은 알고리즘으로 여러 개 분류기를 만들어서 최종 결정하는 알고리즘. 랜덤 포레스트는 그 알고리즘을 결정 트리로 통일 함!

데이터 세트는 전체 데이터 세트에서 중첩을 허용해서 샘플링해 학습함. 이걸 bootstrapping이라고 함.

`RandomForestClassifier` 클래스를 이용해 예측.

랜덤 포레스트 하이퍼 파라미터 및 튜닝

결정 트리 기반의 앙상블 알고리즘은 하이퍼 파라미터가 너무 많다는 단점이 있음.

- `n_estimators` : 랜덤 포레스트에서 결정 트리의 개수를 지정합니다. 디폴트는 10개입니다. 많이 설정할수록 좋은 성능을 기대할 수 있지만 계속 증가시킨다고 성능이 무조건 향상되는 것은 아닙니다. 또한 늘릴수록 학습 수행 시간이 오래 걸리는 것도 감안해야 합니다.
- `max_features`는 결정 트리에 사용된 `maxfeatures` 파라미터와 같습니다. `RandomForestClassifier`의 기본 `max_features`는 'None'이 아니라 'auto', 즉 '`sqrt`'와 같습니다. 따라서 랜덤 포레스트의 트리를 분할하는 피처를 참조할 때 전체 피처가 아니라 `sqrt`(전체 피처 개수)만큼 참조합니다(전체 피처가 16개라면 분할을 위해 4개 참조).
- `max_depth`나 `min_samples_leaf`, `min_samples_split`과 같이 결정 트리에서 과적합을 개선하기 위해 사용되는 파라미터가 랜덤 포레스트에도 똑같이 적용될 수 있습니다.