11강. Ensemble Learning 2.

◈ 담당교수 : 김 동 하

■ 학습개요

여러 개의 모형을 결합하여 하나의 최종 예측 모형을 만드는 대표적인 기법인 부스팅에 대해 학습한다. 이진 분류 문제를 해결하는 대표적인 부스팅 알고리즘인 AdaBoost에 대해 깊게 다룰 예정이며, AdaBoost 알고리즘을 경험위험함수의 최소화 관점에서 해석하는 방법에 대해서도 배운다.

■ 학습목표

1	부스팅의 개념에 대해 학습한다.
2	대표적인 부스팅 알고리즘인 AdaBoost에 대해 학습한다.
3	AdaBoost 알고리즘을 어떻게 해석할 수 있는지 학습한다.

■ 주요용어

용어	해설
부스팅	예측력이 약한 모형들을 결합하여 강한 예측모형을 만드는 대표적인 앙상블 기법 중에 하나.
AdaBoost	최초의 부스팅 알고리즘으로 이진 분류 문제를 해결하기 위해 고안되었으며 해당 시점에서 직전의 학습기가 오분류하는 자료를 특별히 더 잘 맞추도록 학습기가 만들어진다.
경험위험함수	학습 자료와 손실 함수를 이용해 만든 목적 함수.
가파른 강하 알고리즘	데이터를 가장 잘 학습하는 방향으로 모형을 반복적으로 학습해나가는 알고리즘. 대표적으로 기울기 강하 알고리즘, 뉴턴-랩슨 알고리즘 등이 있다.

■ 학습하기

01. 부스팅 개요

부스팅의 기본 아이디어

- 예측력이 약한 모형 (weak learner) 들을 결합하여 강한 예측모형을 만드는 것.
- Weak learner
  -> 랜덤하게 예측하는 것보다 약간 좋은 예측력을 지닌 모형.
  -> 반면, 강한 예측모형은 예측력이 최적에 가까운 예측모형을 뜻함.

예제

- 경마에서 우승하는 말을 맞추기 위한 전략
- 여러 전문가들로부터 기본적인 전략을 들을 수 있음.
  -> 최근에 가장 많이 우승한 말에게 배팅
  -> 주변 사람이 가장 선호하는 말에 배팅
- 이들은 임의로 예측하는 것보다는 좋은 전략이지만 정확성이 높지는 않음.
- 이들의 전략을 모두 모아서 결합한다면 높은 예측력을 발휘할 수 있을 것.
- 각각의 전략을 합칠 때, 가중치를 어떻게 부여할 것인가?

02. AdaBoost

부스팅의 기본 아이디어

- Freund and Schapire (1997)
- 최초의 부스팅 알고리즘
- 이진 분류 문제를 푸는 부스팅 알고리즘.
  -> $y \in \{-1,1\}$를 가정.

AdaBoost 알고리즘

1. n개의 가중치를 $w_i = \frac{1}{n}, i = 1,...,n$로 초기화.
2. $m = 1,...,M$에 대해서 다음의 과정을 반복
   ① 가중치 w_i를 이용하여 이진 분류기 $f_m(x)$를 적합한다.
   ② err_m를 다음과 같이 계산한다:

$$err_m = \frac{\sum_{i=1}^{n} w_i \cdot I(y_i \neq f_m(x_i))}{\sum_{i=1}^{n} w_i}$$

③ $c_m = \log\left(\frac{1-err_m}{err_m}\right)$로 설정.

④가중치 w_i를 다음과 같이 업데이트한다:

$$w_i \leftarrow w_i \cdot \exp(c_m \cdot I(y_i \neq f_m(x_i)))$$

3. 단계2에서 얻은 M개의 분류기를 결합하여 다음과 같은 최종 분류기를 얻는다:

$$f(x) = sign\left(\sum_{m=1}^{M} c_m \cdot f_m(x)\right)$$

AdaBoost의 해석 1.

- 단계 2의 ③, ④가 매우 중요함.
- 예측 모형 $f_m(x)$가 랜덤한 추측보다 조금 더 좋은 예측력을 갖는다고 가정하자.
  -> $f_m(x)$의 (가중) 오분류율인 err_m은 0.5보다 작게 됨.
  -> 따라서, c_m은 0보다 큰 값을 가짐.
- ④에서 다음 단계에 사용될 관측치가 다음과 같은 방식으로 할당됨.
  -> 정분류된 관측치의 가중치는 기존의 값과 같음.
  -> 오분류된 관측치의 가중치는 증가.
- 다시 말해서 AdaBoost 알고리즘은
  -> 매 반복마다 오분류된 관측치의 가중치는 증가시키고,
  -> 정분류된 가중치는 감소시키면서 해당 시점의 예측모형을 만들어감.

AdaBoost의 weak learner

- AdaBoost는 의사결정나무를 weak learner로 사용
- Bagging이나 Random Forest와는 달리 그루터기(stump)를 많이 사용한다.
  -> 그루터기: 한 번의 분리만을 사용한 의사결정나무

AdaBoost의 해석 2. - 가파른 강하 알고리즘

- AdaBoost 는 지수손실함수에 대한 경험 위험 최소 추정량을 가파른 강하 알고리즘을 사용해 추정한 결과라는 사실이 증명됨.
- 지수 손실 함수:

$$L(y, f(x)) = \exp(-y \cdot f(x))$$

- 지수손실함수를 이용한 경험위험함수:

$$\frac{1}{n}\sum_{i=1}^{n}\exp(-y_i \cdot f(x_i))$$

- 가파른 강하 알고리즘: 다음과 같은 형태로 업데이트되는 알고리즘.

$$u^{(t+1)} = u^{(t)} + \alpha^{(t+1)} \cdot \Delta u^{(t+1)}$$

  -> 예: Gradient descent algorithm, Newton-Rhapson algorithm
- 지수 손실 함수를 이용한 경험 위험 함수를 최소화하는 함수를 찾고자 함.
- m번 반복했을 때의 함수를 $F_m(x)$라 하면, $(m+1)$번째 반복을 통해 얻는 함수는 다음의 결과가 될 것임:

$$F_m(x) + \alpha_{m+1} \cdot f_{m+1}(x)$$

- 주어진 $F_m(x)$ 하에서 경험 위험 함수를 최소화하는 $\alpha_{m+1}, f_{m+1}(x)$를 찾는 것이 목표.
- 즉,

$$\alpha_{m+1}, f_{m+1}(x) = argmin_{\alpha, f} \sum_{i=1}^{n} \exp\left(-y_i \cdot \left[F_m(x_i) + \alpha \cdot f(x_i)\right]\right)$$

- 최적의 $f_{m+1}(x)$는 $\alpha = 1$를 가정하고 계산해도 상관없음:

$$\begin{aligned} f_{m+1}(x) &= argmin_f \sum_{i=1}^{n} \exp\left(-y_i \cdot \left[F_m(x_i) + f(x_i)\right]\right) \\ &= argmin_f \sum_{i=1}^{n} w_i \cdot \exp\left(-y_i \cdot f(x_i)\right) \end{aligned}$$

-> 여기서, $w_i = \exp\left(-y_i \cdot F_m(x_i)\right), i = 1, ..., n$.

- $f_{m+1}(x)$가 주어졌을 때 경험 위험 함수를 최소화하는 α는 다음과 같음:

$$\begin{aligned} \alpha_{m+1} &= argmin_\alpha \sum_{i=1}^{n} \exp\left(-y_i \cdot \left[F_m(x_i) + \alpha \cdot f_{m+1}(x_i)\right]\right) \\ &= argmin_\alpha \sum_{i=1}^{n} w_i \cdot \exp\left(-y_i \cdot \alpha \cdot f_{m+1}(x_i)\right) \\ &= argmin_\alpha \sum_{i=1}^{n} \left[w_i \cdot \exp(-\alpha) \cdot I(y_i = f_m(x_i))\right] \\ &\quad + \sum_{i=1}^{n} \left[w_i \cdot \exp(\alpha) \cdot I(y_i \neq f_m(x_i))\right] \end{aligned}$$

- 앞의 식을 최소화하는 α를 찾으면 다음과 같다:

$$\alpha_{m+1} = \frac{1}{2} log\left(\frac{1 - err_{m+1}}{err_{m+1}}\right)$$

- 가파른 강하 알고리즘으로 구한 최종 함수와 AdaBoost 알고리즘으로 구한 함수는 왜 같은가?
  -> 0.5를 곱해도 최종 함수값의 부호는 같기 때문.

AdaBoost의 추가 설명

- AdaBoost 알고리즘의 본래 목적은 훈련오차를 빨리 그리고 쉽게 줄이는 것.
- 그러나 AdaBoost 알고리즘이 제안된 이후 본래의 목적과는 상관없이 예측오차도 감소한다는 것이 경험적으로 증명.
- AdaBoost 알고리즘은 배깅과는 다르게 간단한 의사결정나무인 그루터기 (stump)를 많이 사용.

다양한 부스팅 알고리즘

- RealBoost
  -> Schapire and Singer (1999)
  -> AdaBoost 를 변형하여 회귀 문제에 응용한 알고리즘.
- Gradient Boosting

-> Friedman (2001)
-> 부스팅 알고리즘을 기울기 강하 알고리즘으로 해석하여 이를 확장한 알고리즘.
-> 분류, 회귀 등 모든 문제에 사용 가능한 알고리즘.
-> 가장 많이 사용되는 부스팅 알고리즘.

03. Python을 이용한 실습

데이터 설명

- 보스턴 주택 가격 데이터
  -> 1978년에 발표된 데이터로 미국 보스턴 지역의 주택 가격에 영향을 미치는 요소들을 정리
  -> 총 13가지의 요소들과 주택 가격으로 이루어져 있음.
- 부스팅 기법을 이용하여 주택 가격을 예측하는 회귀 모형을 만들자.

```
import pandas as pd
import numpy as np
import os
from sklearn.datasets import load_boston
from sklearn.model_selection import train_test_split

from sklearn.metrics import confusion_matrix
from sklearn.ensemble import GradientBoostingRegressor
#from sklearn.ensemble import GradientBoostingClassifier

import matplotlib.pyplot as plt
```

- 데이터 불러오기

```
boston = load_boston()

X_boston = pd.DataFrame(boston.data, columns=boston.feature_names)
y_boston = pd.DataFrame(boston.target, columns=['MEDV']).iloc[:,0]
```

```
X_train, X_test, y_train, y_test = \
train_test_split(X_boston, y_boston,
                 test_size = 0.3, random_state=123)
```

부스팅 적합하기

- 1,000개의 weak learner 사용.
- 각각의 weak learner는 최대 깊이 3의 의사결정나무를 사용.

```
gbm_model = GradientBoostingRegressor(n_estimators = 1000,
                                      max_depth = 3)

gbm_model.fit(X_train, y_train)
```

- 시험 데이터에서의 성능 확인하기

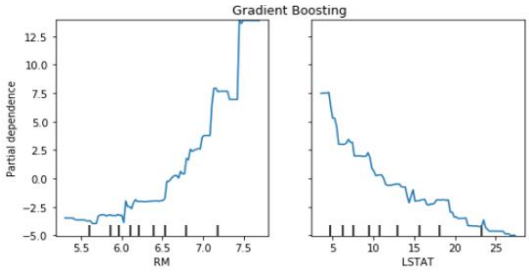
```
gbm_pred = gbm_model.predict(X_test)
print((y_test-gbm_pred).pow(2).mean())
```

```
13.37238327786511
```

모형 결과 시각화하기

- 부분 의존성 그림 그리기

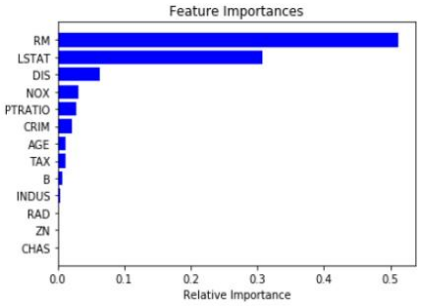
```
from sklearn.inspection import plot_partial_dependence
fig, ax = plt.subplots(figsize=(8, 4))
ax.set_title("Gradient Boosting", fontsize=12)
tree_disp = plot_partial_dependence(gbm_model, X_train, ["RM","LSTAT"], ax=ax)
```



- 변수 중요도 그림 그리기.

```
importances = gbm_model.feature_importances_
indices = np.argsort(importances)

plt.title('Feature Importances')
plt.barh(range(len(indices)), importances[indices], color='b', align='center')
plt.yticks(range(len(indices)), [boston.feature_names[i] for i in indices])
plt.xlabel('Relative Importance')
plt.show()
```



■ 연습문제

(O/X)1. AdaBoost는 배깅, 랜덤 포레스트와 마찬가지로 붓스트랩 자료를 활용한다.

정답 : X

해설 : AdaBoost는 붓스트랩 자료를 활용하지 않는다.

(O/X)2. AdaBoost는 매시점마다 직전 시점의 학습기와는 무관하게 독립적으로 만들어진다.

정답 : X

해설 : AdaBoost는 매시점마다 직전 시점의 학습기가 오분류한 관측치를 더 잘 맞추도록 학습되어진다.

(객관식)3. AdaBoost는 OO 손실함수를 이용한 경험위험함수를 최소화하는 방법론으로 이해할 수 있다. 어떤 손실함수인지 적으시오.

정답 : 지수손실함수

해설 : AdaBoost는 지수손실함수를 이용한 경험위험함수를 가파른 강하 알고리즘을 통해 최소화하는 방법론으로 해석할 수 있다.

■ 정리하기

1. 부스팅은 배깅, 랜덤 포레스트와 마찬가지로 약한 학습기 여러 개를 결합하여 예측력이 강한 하나의 모형을 구축하는 방법론이다.
2. 대표적인 부스팅 알고리즘인 AdaBoost는 이진 분류 문제를 풀기 위해 고안되었으며, 매 반복마다 직전 학습기가 오분류한 관측치를 특별히 더 잘 맞추도록 학습기가 만들어진다.
3. AdaBoost는 가파른 강하 알고리즘을 이용해 지수손실함수를 이용한 경험위험함수를 최소화하는 방식으로 이해할 수 있다.

■ 참고자료 (참고도서, 참고논문, 참고사이트 등)

박창이, 김용대, 김진석, 송종우, 최호식. 『R을 이용한 데이터마이닝』. 서울:교우사, 2018.