Boosting

<Boosting 기법의 종류>

- Ada Boosting(Adaptive Boosting)
- Gradient Boosting
- XG Boosting(Extreme Gradient Boosting)

● Ada Boosting

<개념>

Ada Boosting은 Adaptive Boosting의 약자로 성능을 향상시키기 위하여 다른 많은 형태의 학습 알고리즘과 결합하여 사용할 수 있다. AdaBoost는 이전의 분류기에 의해 잘못 분류된 것들을 이어지는 약한 학습기들이 수정해줄 수 있다는 점에서 다양한 상황에 적용할 수 있다(adaptive). 간단한 약분류기(weak classifier ex.small decision trees)들이 상호보완 하도록 단계적으로 학습하여, 이를 조합하여 만들어진 최종 강분류기(strong classifier)의 성능을 증폭시키는 원리이다.

※Bagging, Boosting 비교

1. Bagging 은 독립적으로 모델을 구성하지만, Boosting 은 이전 모델에서 실패한 것을 바탕으로 새로운 모델을 구성한다

2. 두가지 모두 복원 랜덤 샘플링을 하지만 Boosting 은 학습오류가 큰 데이터샘플에 가중치를 부여한다.

3. Bagging 은 over-fitting 을 해결하는데 도움이 되지만, Boosting 은 bias 를 줄이는데 도움이 된다.

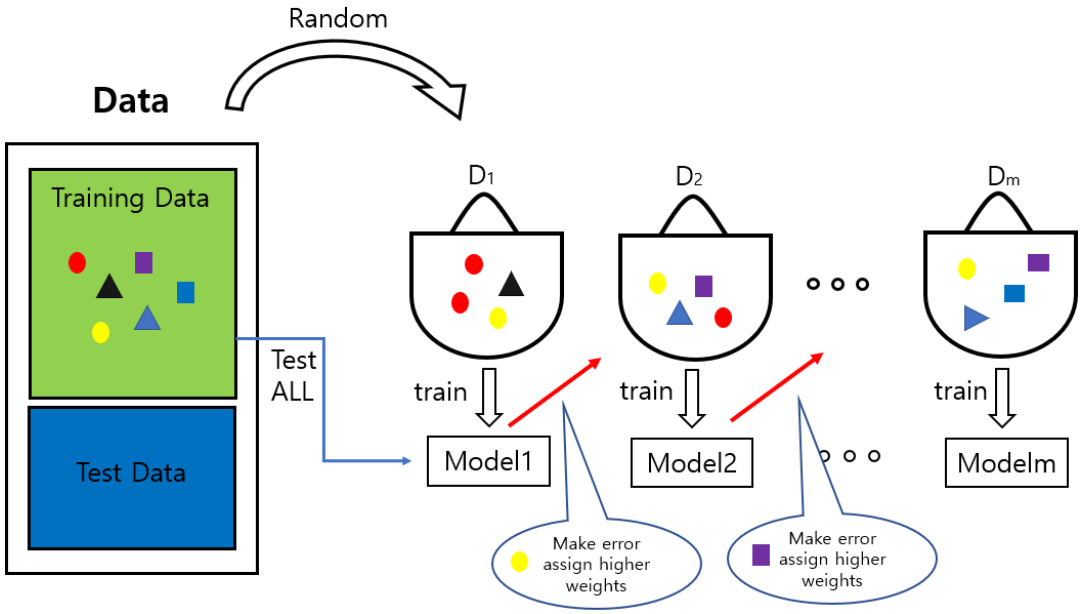
그렇다면 두가지 방법 중 어떤 방법을 선택하는 것이 좋을까? 두가지 방법 모두 여러 모델을 종합하여 결과를 내기 때문에 단일 모델일 때보다 분산을 감소시켜 안정성이 높다. 다만 문제 상황에 따라 bagging을 선택할지 boosting을 선택할지 달라진다. 만약 단일 모델의 성능이 낮은 것이

문제라면 boosting을 선택하는 것이 낫다. Boosting은 단일 모델의 장점을 최적화하고 단일 모델에서 나타날 수 있는 위험을 감소시키기 때문에 오류가 적은 결합 모델을 생성할 수 있기 때문이다. 반대로 단일 모델의 over-fitting이 문제라면 bagging을 선택하는 것이 낫다. Boosting은 오히려 over-fitting을 증가시키기 때문에 문제를 해결하는데 도움이 되지 않는다.

<특징>

- Boosting은 bias줄이려는 것에 초점을 맞춘 기법이므로 과적합(overfitting)의 문제가 발생할 수 있다. 따라서 모델의 모수를 적절하게 조정하는 것이 중요하게 작용한다.
- Bias가 줄어들어 정확도는 올라가지만 outlier에 취약하다.

<작동 원리>



먼저 training data set에서 랜덤으로 데이터를 뽑아 D1을 구성한다. D1의 데이터를 바탕으로 model1을 생성한 뒤 training data set 전체를 사용하여 성능을 테스트한다. 테스트 결과 위의 그림처럼 노란색의 에러가 크게 나왔다면 이는 model1에서 해결하지 못한 문제이기 때문에 가중치를 주어 D2를 구성할 때 뽑힐 확률을 증가시킨다. 즉 다음 model이 그 문제에 집중할 수 있게 하는 것이다. 마찬가지로 가중치가 부여된 training data set에서 D2를 뽑아 model2를 생성하고 테스트한 뒤 결과에 따라 가중치를 부여한다. 이 때 mini

ensemble처럼 model1과 model2의 결과를 종합하여 가중치를 부여한다. 이런 방식으로 이전 model들의 test결과를 반영하여 다음 model을 생성하고, 분류모델의 경우는 weighted majority vote, 회귀모델의 경우에는 weighted sum을 사용하여 최종 모델을 생성한다.

단계별로 나누면

- 1단계) 현재 주어진 데이터셋에 대해 상대적으로 단순한 모델을 이용하여 학습을 진행.
- 2단계) 학습오류가 큰 개체의 선택확률을 증가시키고, 학습 오류가 작은 경우의 데이터샘플의 선택확률을 감소시킨다.
- 3단계) 2단계에서 조정된 확률을 기반으로 다음 단계에서 학습할 데이터 셋을 구성. 이 과정을 통해 현재의 모델이 잘 해결하지 못하는 어려운 데이터 샘플에 집중하게 된다.
- 4단계) 종류조건을 충족하는 오류가 발생할 경우까지 위의 단계를 반복하여 실행
- 5단계) 일반적으로 분류모델의 경우는 weighted majority vote, 회귀모델의 경우에는 weighted sum을 사용하여 최종 모델을 생성한다.

<Ada Boosting 알고리즘 간단 해석>

Algorithm AdaBoost

1: Init data weights $\{w_n\}$ to $1/N$
2: **for** $m = 1$ to M **do**
3: fit a classifier $y_m(x)$ by minimizing weighted error function J_m:
4: $J_m = \sum_{n=1}^{N} w_n^{(m)} 1[y_m(x_n) \neq t_n]$
5: compute $\epsilon_m = \sum_{n=1}^{N} w_n^{(m)} 1[y_m(x_n) \neq t_n] / \sum_{n=1}^{N} w_n^{(m)}$
6: evaluate $\alpha_m = \log\left(\frac{1-\epsilon_m}{\epsilon_m}\right)$
7: update the data weights: $w_n^{(m+1)} = w_n^{(m)} \exp\{\alpha_m 1[y_m(x_n) \neq t_n]\}$
8: **end for**
9: Make predictions using the final model: $Y_M(x) = \text{sign}\left(\sum_{m=1}^{M} \alpha_m y_m(x)\right)$

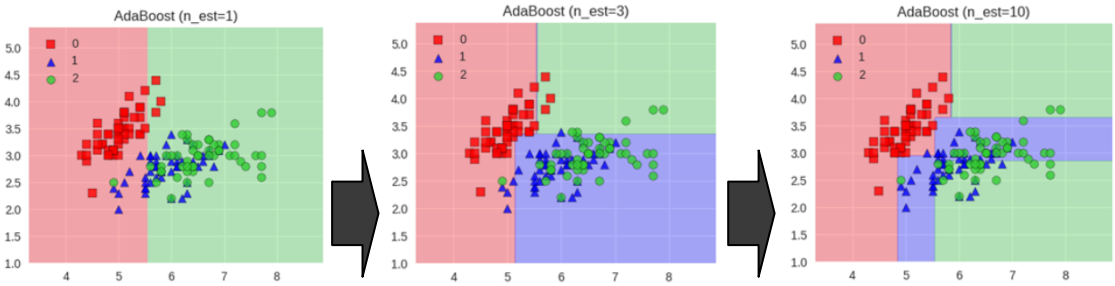
1. 첫 번째 분류기인 y1(x)는 데이터 샘플에 모두 같은 가중치(1/N)를 부여하여 학습.

2. **ε**은 base classifier의 가중치가 부여된 오류를 나타내고, **α**는 데이터에 부여하는 가중치를 의미

한다. 따라서 **ε**가 작을수록 큰 **α**값이 다음 학습할 데이터 샘플의 새로운 가중치로서 적용된다.

3. 이러한 과정의 반복을 통해 잘못 분류한 데이터 샘플에 관하여는 가중치가 증가하고 바르게 분석한 데이터 샘플에 대해서는 가중치가 감소한다.

<Desion tree 모델을 이용한 Ada Boosting 예시>



위의 경우에 사용된 기초 학습자는 depth=1 인 Decision tree. Boosting 기법을 통해 각 단계에서 오분류된 부분에 집중하여 반복하여 데이터 샘플을 재구성하여 Decision tree 알고리즘을 적용.

● Gradient Boosting

<개념>

Gradient Boosting은 경사하강법을 사용해서 Ada Boosting보다 성능을 개선한 Boosting기법이다. Ada Boosting은 높은 가중치를 가진 데이터(outlier)가 존재하게 되면 성능이 크게 떨어질 수 있는 단점이 있다. 높은 가중치를 가진 데이터 포인트 근처의 다른 포인트들이 의도하지 않게 잘못된 분류로 될 가능성이 높은 것이다. 따라서 Gradient Boosting기법에서는 가중치를 계산하는 방식에서 경사하강법을 통해 오류를 최소화하는 최적화 문제로 해법을 찾는다.

<특징>

- 특성의 스케일을 조정하지 않아도 되고(outlier에 덜 민감하게 반응), 연속적인 데이터 특성

에서도 잘 작동한다.

- Ada Boosting에서 보다 학습훈련의 시간이 길게 걸리므로 경사 하강법 적용 시 적절한 변수 조정이 필요하다.
- 고차원 데이터에서는 잘 작동하지 않을 수 있다.
- Regression과 Classification 문제에 모두 적용 가능하다. 단 경우에 따라 손실함수는 달라진다.
- 경사하강법도 지역최소값에 빠질 위험성이 존재하지만 이를 해결하기 위한 다양한 방법이 연구되고 있다.

<Gradient Boosting 알고리즘 간단 해석>

기존 분류기의 오차 발생 - 오차를 손실함수로 표현 - 경사하강법의 적용 - 반복 - 최적의 분류기

$$F_m(x) = F_{m-1}(x) + \gamma_m h_m(x)$$

$$F_m(x) = F_{m-1}(x) + argmin_h \sum_{i=1}^{n} L(y_i, F_{m-1}(x_i) + h(x_i))$$

: 현재의 Fm-1(x)가 주어졌을 때, 경사하강법을 통해 손실함수 L을 최소화하는 Decision tree h(x)를 찾고 그 때의 개선 모델인 Fm(x)를 생성한다. 이 과정을 반복하여 최적모델을 도출한다.

● XG Boositng

<개념>

XG Boosting은 Extreme Gradient Boosting의 약자로 기존의 Gradient Boosting의 속도문제를 해결해야 하기 위해 전산속도와 모델의 성능에 초점을 맞춘 기법이다. 학습성능은 좋지만 연산속도/수행시간 측면에서는 부족한 측면이 많다 . XGBoost는 Decision tree를 구성할 때 병렬처리 기법을 사용해서 수행시간 측면에서 Gradient Boosing 보다 비약적으로 향상되었다.

<특징>

- 기존 Gradient Boosing 비해 연산속도가 빠르고 모델의 성능이 향상된다.
- 과적합(Overfitting)이 잘 일어나지 않는다.
- 훈련하는 동안 컴퓨터의 사용 가능한 모든 CPU를 사용함으로써 트리의 구조를 병렬화 한다.
- 다른 알고리즘과 연계활용성이 좋다.
- 최근의 앙상블 모델중에서 가장 우수한 알고리즘으로 평가받아 각종 대회에서 사용된다.

출처

https://www.youtube.com/watch?v=GM3CDQfQ4sw&feature=youtu.be

https://blog.naver.com/euleekwon?Redirect=Log&logNo=221377873711

https://ko.wikipedia.org/wiki/%EC%97%90%EC%9D%B4%EB%8B%A4%EB%B6%80%EC%8A%A4%ED%8A%B8