Random forest & Gradient boosting

## 의사결정나무

의사결정나무(decision tree) 알고리즘은 기본적으로 설명변수 공간을 다수의 영역으로 분할하는 방법입니다. 해당 관측치가 속하는 영역의 훈련 관측치들의 평균 (회귀) 또는 최빈값 (분류)을 사용하여 예측합니다.

회귀 문제에서는 아래 식으로 주어진 RSS 를 최소로 하는 셜명변수 공간을 찾는 것이 목적입니다.

Rj: jth 설명변수 공간  
: jth 설명변수 공간에 속한 훈련 관측치 반응변수들의 평균값

설명변수 공간을 J개로 분할하기 위해 보통 top-down 방식의 반복적인 이분 분할 전략을 이용합니다(**recursive binary splitting**)

위 전략은 다음 두 가지 단점이 있습니다(*drawbacks*)

* 탐욕적이고 근시안적임(greedy, short-sighted)
* 과적합에 취약(prone to overfitting)

탐욕적이라는 것은 트리를 만드는 과정의 각 단계에서 미리 앞을 내다보고 나중에 오는 어떤 단계에서 더 나은 트리가 될 분할을 선택하는 것이 아니라 그 특정 단계에서 가장 좋은 분할을 선택한다는 뜻입니다.

예를 들어, 5만원짜리 지폐, 3만원짜리 지폐, 5천원짜리 지폐가 있다고 해봅시다. 가장 적은 수의 지폐를 사용하여 6만원을 만들어야 한다고 해보죠. Top-down 방식의 탐욕적 알고리즘은 어떤 답을 내놓을까요?

의사결정나무가 과적합에 취약하다는 것은 어떤 뜻일까요?

## 가지치기

의사결정나무의 단점에 대한 해결방안으로 가지치기(*pruning*) 전략이 있습니다.

**cost complexity pruning (weakest link pruning)**  
모든 가능한 subtree 를 고려하는 대신에 tuning parameter (>=0) 에 의해 색인된 일련의 tree 들을 고려.

각 값에 대해, 아래 식이 최소가 되는 subtree T 를 구할 수 있다.

T: subtree T의 number of terminal nodes  
: tuning parameter

일때, subtree T=  
가 증가함에 따라 많은 터미널 노드가 있는 트리의경우 항이 크게 증가할 것이므로 트리가 작을 때 위 식의 값이 최소로 되는 경향이 있습니다.

즉, tuning parameter 는 서브트리의 복잡도와 훈련자료에 대한 적합 사이의 trade-off 를 제어합니다.

k-fold CV 을 이용해서 검정오차를 의함수로 평가하고,평균 검정오차를최소로하는 를 선택합니다.

Linear model vs. Tree-based model

## 분류 오차

분류 문제에서 의사결정나무는 아래의 분류 오류율(E, classification error, misclassification rate)을 최소로 하는 것이 목적입니다.

: m번째 설명변수 공간 내 k class에 속하는 관측치들의 비율

이분 분할(binary split) 과정에서 위 식의 분류 오류율을 사용할 수 있지만, 실제로는 node purity에 더 민감한 아래 두 가지 척도를 주로 사용합니다.

* 지니 지수(Gini index)
* 교차엔트로피(cross-entropy)

**Gini index**

**Cross-entropy**

## Bagging

배깅(bagging)은 bootstrap aggregation으로도 알려져 있고, 기계학습모델의 분산을 줄여 예측 정확도를 증가시키기 위한 범용 절차(general-purpose procedure)입니다.

회귀에서는 평균값을(average for regression), 분류에서는 다수 모델의 예측치에 따라 결정됩니다(majority rule for classification)

배깅에 사용되지 않은 관측치들을 Out-of-bag (OOB) 관측치라고 합니다.

**OOB 오차**란 i번째 관측치에 대해 그 관측치가 OOB 이었던 각각의 트리를 이용하여 검정 오차를 추정할 수 있습니다. 교차검증 또는 검증셋 기법을 수행하기 힘든 규모가 큰 데이터셋에 대해 특히 편리합니다.

**Variable importance** 주어진 설명변수에 대한 분할로 인한 검정 오차(RSS or Gini index/Cross entropy)의 감소분을 모든(B개) 트리에 대해 평균합니다. 이 값이 크면 해당 설명변수가 중요하다고 할 수 있습니다.

## 랜덤 포레스트(Random Forest)

참고 자료

* [Random Forest part I](https://www.youtube.com/watch?v=J4Wdy0Wc_xQ)
* [Random Forest part II](https://www.youtube.com/watch?v=nyxTdL_4Q-Q)

배깅에서와 마찬가지로 bootstrap에 의해 다수의 트리를 만드는 것은 동일합니다. 또한, 트리 내에서 split이 고려될 때마다 p개 설명변수 중에서 무작위로 m개의 설명변수가 선택됩니다(, rule of thumb)

위와 같이 설명 변수 공간을 임의로 줄이는 것은 트리들 간의 상관성을 낮추어 결국 분산을 감소시키는 방법입니다.

## 부스팅(Boosting)

여러개의 decision tree 를 만들는 것은 bagging과 동일하다. 그러나, bagging과 달리 훈련셋 자료에 대해 bootstrap 샘플링을 하지 않고, 모든 훈련셋 자료를 이용하여 순차적으로 천천히 학습한다.

회귀 문제에서의 알고리즘

1. 이라하고, 훈련셋의 모든 i에대해 로 설정한다. (r: residuals)
2. b = 1,2,…,B 에 대하여 다음을 반복한다.

* d개의 분할(d+1 터미널 노드)을 가진 트리 를 훈련자료 (X,r)에 적합한다.
* 새로운 트리의 수축 버전(수축 parameter 를 곱한)을 더하여 를 업데이트한다.
* 잔차들을 업데이트한다.

1. 부스팅 모델을 출력한다.

Boosting 의 tuning parameters

* B: number of trees
* : 수축 파라미터 (학습 속도를 제어)
* d: number of split in each tree (boosting의 복잡도를 제어)