9. Bagging & Random Forest
2019년 가을 학기

2019년 08월 23일
https://www.github.com/KU-BIG/KUBIG_2019_Autumn

앞선 강의에서 decision tree에 대해서 살펴보았다. Decision tree는 성능이 비교적 떨어진다는 단점이 있는데, 이번 강의에서는 decision tree 뿐만 아니라 여러 classify들의 성능을 획기적으로 높이는 ensemble기법과 그 중에서도 bagging과 random forest에 대해서 알아보도록 하자.

1. Introduction

Bagging과 Random Forest는 ensemble의 대표적인 기법이다. 음악에서 앙상블은 2인 이상이 하는 노래나 연주를 말한다. 이를 통해서 우리는 ensemble이라는 기법이 2개 이상의 모델을 전체적으로 어울리게, 조화롭게, 통일되게 사용할 것이라는 것을 짐작할 수 있다.

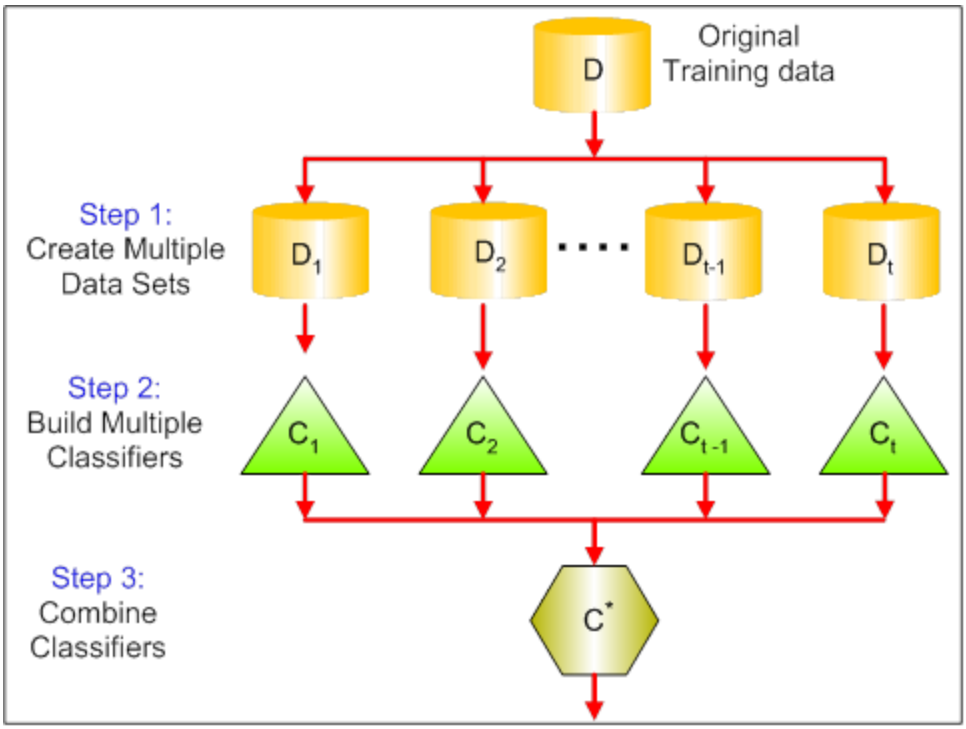
그렇다면 왜 우리는 굳이 2개 이상의 모델을 섞어가면서 ensemble을 활용하는 것일까. 서두에서 설명했듯이 단일의 분류기(classifier)로는 우리가 원하는 성능이 나오지 않기 때문이다. 단일의 성능이 떨어진다면 여러가지로 동시에 활용하면 되지 않을까라는 의문에서 나온 것이 바로 ensemble기법이다. Ensemble은 decision tree에만 활용되지 않고 다양한 model에 활용된다.

음악에서 앙상블에는 다양한 종류의 음악이 있듯이 머신 러닝의 ensemble에도 다양한 종류가 있다. 그 중에서도 이번 강의에서는 bagging과 random forest를 배운다. 그렇다면 bagging이란 무엇인가. Bagging은 한 표본에서 또다른 세부 표본을 추출한 다음에 각 세부 표본에 대해 분류를 진행하는 것이다. 각 분류기에 나온 결과들을 다수결 또는 평균을 통해서 최종 결과를 결정한다.

Random Forest는 ensemble의 최종 진화형이라고 볼 수 있을 정도로 그 성능이 매우 뛰어나다. 이름 그대로 무작위로 숲을 형성하는 기법이기 때문이다. Random한 변수로 decision tree를 형성한 뒤 무작위로 그 결과를 선택한다. 그래서 Random과 Forest가 동시에 이름에 들어간다.

Boosting도 ensemble 중 하나이다. Boosting은 bagging과 비슷하지만, bagging과는 다르게 error가 나온 point에 집중한다. Boosting은 다양한 종류가 있어서 다음 강의에서 다루겠다.

2. Ensemble



https://www.analyticsvidhya.com/wp-content/uploads/2015/07/bagging.png

Ensemble이란 주어진 자료로부터 여러 개의 예측 모형을 만든 후 예측 모형들을 종합하여 하나의 최종 예측 모형을 만드는 방법이다. 이를 통해 단일 모형으로는 부족한 예측력을 향상시킬 수 있다. 이에 대해서는 아래의 예시를 보면서 살펴보자.

25개의 classifier가 있다고 가정하자. 각 분류기들은 서로 상호독립적이며 각각의 분류기의 오분류율(ε)은 0.35라고 하자. Ensemble 분류기가 기본 분류기의 예측에 대해 다수결로 test data의 class label를 예측하는 경우, ensemble 분류기가 잘못된 예측을 할 확률은 다음과 같다.

$$\epsilon_{\text{ensemble}} = \sum_{i=13}^{25} \binom{25}{i} \epsilon^i (1-\epsilon)^{25-i} = 0.06, \quad \epsilon = 0.35$$

각각의 분류기가 제대로 분류를 했는지는 Bernoulli 시행에 해당하므로 위의 확률은 그 합인 Binomial에 의거해서 계산된다. 위의 결과를 봤을 때 하나의 분류기의 정확도는 65%였으나, 25개의 동일한 분류기를 동시에 활용하면서 정확도가 94%까지 오른 것을 알 수 있다.

Ensemble은 위와 같이 예측력을 개선한다는 장점 이외의 장점도 있다.

1) Bias를 감소시킨다. 치우침이 있는 여러 model의 평균을 취하면, 어느 쪽에도 치우치지 않는 결과인 평균을 얻게 된다.
2) Variance를 감소시킨다. 한 개 모형으로부터의 단일 결과보다 여러 모형의 결과를 결합하기에 분산이 작아진다. 이는 표본평균의 분산이 모평균의 분산보다 작다는 것과 비슷한 원리이다.
3) Overfitting(과적합)의 가능성을 줄여준다. Overfitting이 없는 각 모형으로부터 예측을 결합 (평균, 가중 평균, 로지스틱 회귀 등)하면 과적합의 여지가 줄어든다.

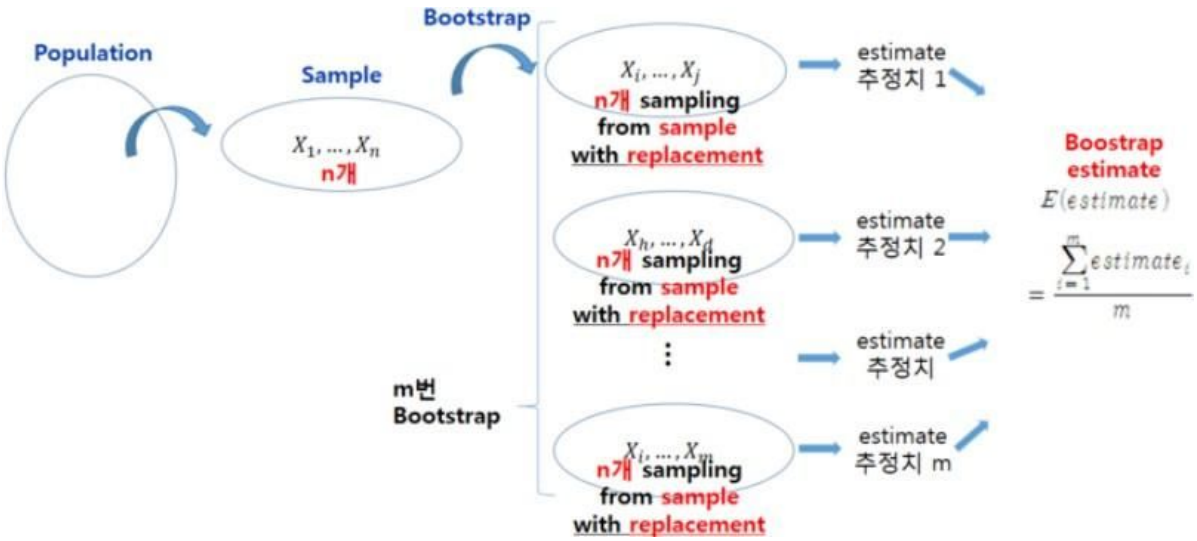
반면에 Ensemble은 여러 단점을 지닌다.

1) 우선 여러 모형의 결과를 종합하여 활용하기에 해석하기에 어려움이 있다. 어느 하나의 모형에서의 결과를 가져오지 않기 때문이다.
2) 이에 따라 모형의 투명성이 떨어지게 되어 현상에 대한 원인을 분석할 때는 적합하지 않다. Decision tree의 경우 parent node에서부터 child node까지의 path로 원인과 결과를 해석할 수 있다. 하지만 tree들이 여러 개 섞이게 되면 더 이상 원인과 결과를 해석할 수가 없다. 단지, 이러한 결과가 다수결에 의해서 선택되었다라고 할 수 밖에 없다.
3) 마지막으로 단일 모델에 비해 예측 시간이 많이 걸린다. 따라서 급히 분석 결과를 내야하는 상황에서는 부적절하며 컴퓨터의 성능에 따라서 해당 모형이 작동하지 않는 상황이 발생할 수도 있다.

Ensemble의 종류는 다음과 같다.

- 데이터를 조정하는 방법 : Bagging, Boosting
- 변수의 개수를 조정하는 방법 : Random Forest
- 집단명을 조정하는 방법
- 분류모형의 가정을 조정하는 방법 : Stacking

3. Bootstrap



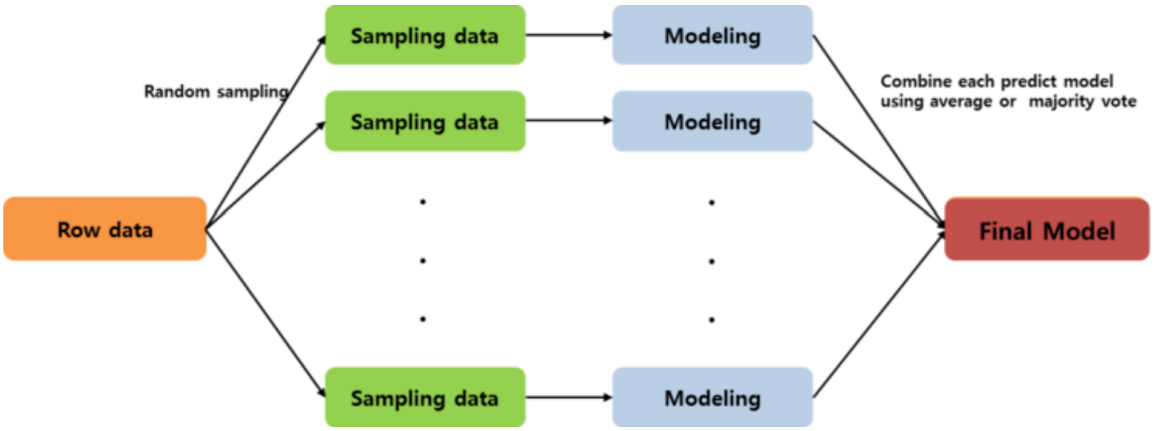
https://m.blog.naver.com/wjddudwo209/220013117867

Ensemble의 기본인 bootstrap에 대해서 간단히 설명하겠다. 주어진 sample(data set)을 원래의 모집단을 대표하는 독립 표본으로 가정하고, 그 자료로부터 중복을 허용한 무작위 추출로 복수의 자료를 작성하고 각각에서 얻어진 통계량을 계산하는 방법이다. 표본으로 모집단을 추정하는 것처럼 표본에서 다시 재추출한 표본으로 표본을 추정하는 아이디어이다.

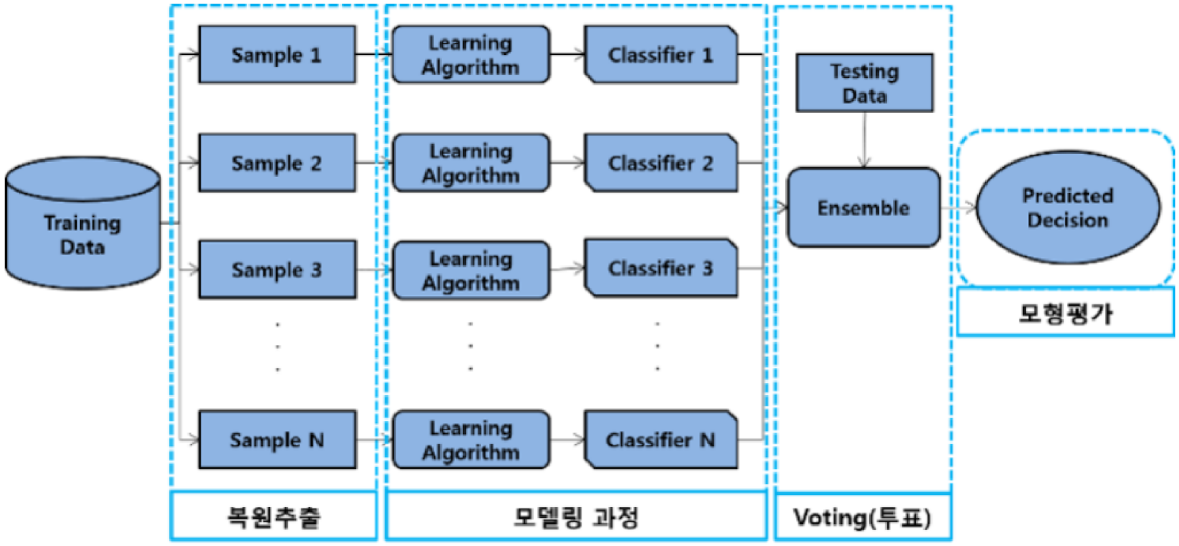
표본에서 또 다른 표본을 재추출할 때 반드시 복원 추출을 해야한다. 복원 추출(Sampling without replacement)을 함으로써 각 bootstrap 표본 원소의 개수는 달라지게 된다. 복원 추출을 하기 때문에 중복된 요소가 뽑힐 수 있으며, 중복된 요소는 제거해야하기 때문이다. Bootstrap인 이렇게 복원추출을 통해서 bootstrap sample을 무수히 많이 생성할 수 있으며, ensemble 역시 이 점을 이용하여 추가적인 model들을 형성한다.

Bootstrap을 활용한 추정량은 다른 추정량보다 bais 및 variance가 작다. 또한 분포의 가정이 없이도 여러 통계량에 대한 p-value를 추출할 수 있기에 가설 검정이 가능하다. 기존의 표본에서 산출된 통계량을 각각의 boostrap 표본들에서 산출된 통계량의 분포와 비교하면 되기 때문이다. 통계학에서 가장 구하기 어려운 standard error의 분포와 confidence Interval 또한 쉽게 구할 수 있다. Standard error를 구할 수 없는 Ridge, Lasso등의 regularized regression은 bootstrap 방법을 통해서 이를 구하게 된다.

4. Bagging



https://greeksharifa.github.io/public/img/Machine_Learning/2018-11-06-TripleB/01.jpg



https://bit.ly/33T6839

Bagging은 bootstrap aggregating의 줄임말로 원 데이터 집합으로부터 크기가 같은 표본을 여러 번 단순임의 복원추출하여 각 bootstrap 표본에 대해 분류기(classifiers)를 생성한 후 그 결과를 결합한다. 주로 decision tree에서 많이 쓰이지만 어떤 알고리즘에 관해서도 사용 가능하다.

최적의 tree를 구축할 때 가장 어려운 부분이 가지치기(pruning)이지만 bagging에선 이를 하지 않고 최대로 성장한 fully grown decision tree를 활용한다. 그래서 각 tree의 분산(variance)은 크고 편향(bias)은 작다. 이러한 tree들을 합침으로서 분산은 줄어들게 된다. $Z_1, \dots, Z_n$의 n개의 독립적인 관측치가 있을 때 각각의 분산이 σ^2이라 하면 $\bar{Z}$의 분산은 σ^2/n이 되는 것과 비슷한 원리라고 생각하면 된다.

4.1 Bootstrapping in Bagging

Bagging은 n개의 원 데이터(Raw data, training data)로부터 B번의 랜덤복원추출(bootstrap)을 하고 각 샘플의 모델링을 통해 나온 예측변수들을 결합하여 최종 모델을 생성한다. 앞서 말했듯이 bootstrap sample은 random sample without replacement이기에 각 시행마다 bootstrap sample에 포함된 중복되지 않은 데이터의 개수는 다르다.

만약에 원 데이터에 n개의 데이터가 있다고 할 때, n의 크기를 가지는 bootstrap sample이 평균적으로 원 데이터의 몇 퍼센트인지 파악하는 것은 이후에 test data를 구성하는데 필요하다. 한 번 n개의 복원 추출 할 때마다 한 번이라도 뽑히는 데이터의 비율은 보통 63.2%이다. 데이터가 충분히 큰 경우 어떤 데이터가 하나의 bootstrap 표본에서 제외될 확률은 다음과 같다.

$$\lim_{n\to\infty}(1-\frac{1}{n})^n = e^{-1} = 0.3678$$

n개의 sample 중 첫 번째 원소가 뽑히지 않을 확률은 $1-\frac{1}{n}$이다. n번의 복원 추출에서 이 첫 번째 원소가 모두 뽑히지 않아야하므로 앞의 확률에 n을 제곱한다. $(1-\frac{1}{n})^n$. 만약에 n이 무수히 커진다면, 우리가 다룰 원 데이터가 충분히 크다면 극한으로 보낼 수 있기에 위와 같이 e^{-1}을 확률로 가지게 된다. 따라서 어떤 데이터가 하나의 bootstrap 표본으로 뽑힐 확률은 위의 경우의 여집합이므로 그 확률은 $1-(1-\frac{1}{n})^n$이 된다.

4.2 OOB

위에서 구한 확률 $1-(1-\frac{1}{n})^n$를 극한으로 보내게 되면 $1-e^{-1}=0.632$라는 확률값을 가지게 된다. 이는 어떤 데이터가 하나의 bootstrap 표본으로 뽑힐 확률이다. n개의 데이터를 복원 추출로 뽑을 때 뽑히지 않은 것의 개수만큼, 대략 36.7%의 중복된 데이터가 존재하게 된다. OOB (Out-of-Bag)는 bootstrap 표본으로 뽑히지 않은 36.7%의 데이터를 의미한다.

보통의 머신러닝을 수행할 때는 train data와 test data를 따로 쪼개서 train data에 모형을 적합시키고 test data에 적합한 모형을 적용하여 실제 모형이 어느 정도의 성능을 가지고 있는지 확인한다. 하지만 bootstrap을 이용하면 test data를 구성할 필요 없이 OOB 데이터를 이용해서 test error를 구하거나 변수의 중요도를 측정할 수 있다. 알아서 bootstrap이 모델을 적합하는데 사용된 데이터(bootsrap sample로 뽑힌 데이터)와 그렇지 않은 데이터를 걸러주기 때문이다.

어떠한 데이터가 OOB가 될 확률이 약 1/3이므로 model을 test하는데에는 약 B/3개의 데이터들을 사용하게 된다. Model을 test하는 것은 앞서 배운 것과 동일하게 해당 관측값이 OOB로 분류된 데이터들을 인풋으로 넣은 후에 각 결과들을 평균내면 된다. Test에 이용되는 관측치(OOB)로 학습하지 않았으므로 test 결과는 유효하다. 만약 데이터 개수가 충분히 많다면 OOB error는 Leave One Out Cross Validation error와 같게 된다.

4.3 Aggregating in Bagging

Bootstrap을 통해서 각 bag마다 모델을 적용하여 예측값들을 계산하였다면 이들을 모두 결합하여 최종적인 결론을 내야한다. 예측값들을 결합하는 방법은 변수가 연속형일 경우에는 평균, 범주형일 경우에는 다수결(majority vote)을 사용하는 것이 일반적이다.

Pseudo Algorithm of Bagging

1. **K** = size of bootstrap sample, **n** = size of sample
2. **for k** = 1 to **K** do
    a. Make **n** size of bootstrap sample D_k
    b. Build a model C_k on Bootstrap sample D_k
3. **end** for loop
4. **Aggregate** the result of C_k s.

4. 번에서 C_k 의 결과들을 합산하는 것은 다음의 수식을 활용하면 된다.

$\text{Regression} : C^* = \frac{1}{K}\sum_{k=1}^{K} C_k$, 각 모델의 평균

$\text{Classification} : C^*(x) = \text{argmax}_y \sum_{k=1}^{K} I(C_k(x) = y)$, Majority vote (다수결)

4.4 Bagging with example

마지막으로 위에서 배운 내용들을 예시를 통해서 간단하게 복습한 뒤에 bagging의 장단점을 간단하게 정리한 후 다음 장, random forest로 넘어가고자 한다.

Original Data	1	2	3	4	5	6	7	8	9	10
Bag1 D_1	7	8	10	~~8~~	2	5	~~10~~	~~10~~	~~5~~	9
Bag2 D_2	1	4	9	~~1~~	2	3	~~2~~	7	~~3~~	~~2~~
Bag3 D_3	1	8	5	10	~~5~~	~~5~~	9	6	3	7

위는 각 Bag마다 10의 크기를 가지는 bootstrap sample이 담겨 있다. Bag1, Bag2, Bag3에는 각각 6, 6, 8개의 고유한 data가 담겨 있다. Bag1에서의 OOB는 1, 3, 4, 6, Bag2에서의 OOB는 5, 6, 8, 10, Bag3에서의 OOB는 2, 4이다.

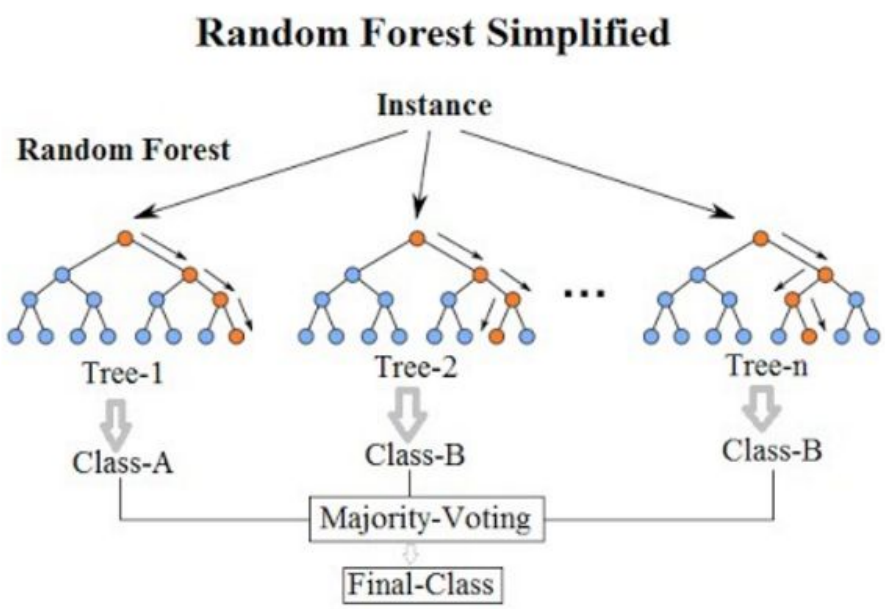
각 모델들의 결과를 합산하는 과정을 보자. Bag1의 bootstrap sample 2, 5, 7, 8, 9, 10으로 model C_1을 적합시키고 예측 값 y_1을 계산한다. Bag2, Bag3도 Bag1과 같이 bootstrap sample로 각각 C_2, C_3를 적합시키고 예측 값 y_2, y_3을 계산한다. Regression이라면 y_1, y_2, y_3의 평균을 구하고 classification이라면 다수결을 통해 최종 결과물을 계산한다.

마지막으로 OOB error를 구하는 것을 살펴보자. Bag1에 적용된 model C_1에 OOB 데이터인 1, 3, 4, 6을 input으로 넣어 output을 산출한 뒤 error를 계산한다. Bag2와 Bag3도 Bag1와 같이 OOB data를 각각 C_2, C_3에 넣어 test error를 계산한다. 그 후 계산된 error의 평균을 활용하면 test error로 활용할 수 있는 OOB error가 나오게 된다.

Bagging의 장점은 다음과 같다. 결합된 모형의 기대 손실이 단일 모형의 기대손실보다 항상 작다. Variance를 줄여줌으로써 예측력을 향상시켜 과적합을 방지한다. Bagging의 단점은 다음과 같다. Tree에서 활용되는 독립변수 선정에 대한 고민 없이 무작정 decision tree를 양산하기 때문에 tree간 높은 상관성이 생기게 된다. 우리는 다음 장에서 이러한 문제를 해결하는 Random Forest에 대해서 배우게 될 것이다.

5. Random Forest

2001년에 Leo Breiman에 의해 처음으로 소개된 기법으로 decision tree의 단점을 개선하기 위한 알고리즘 중 가장 지배적인 알고리즘 중 하나이다. 다수의 decision tree를 결합하여 하나의 모형을 생성하는 방법이다. Random Subspace method Feature Bagging이라고도 부른다. 기계학습에서 변수를 전부 이용하는 것 대신 몇 가지를 임의 추출하여 이용하는 것을 의미한다.



https://miro.medium.com/max/1184/1*i0o8mjFfCn-uD79-F1Cqkw.png

Bagging과 Random Subspace method 개념을 결합해 고안한 기법이 Random Forest이다. Bagging과의 차이는 임의성(randomness)를 관측치뿐만 아니라 변수에도 적용했다는 것이다. 변수도 임의로 추출(Random Subspace)하기 때문에 변수간 상관성이 높은 변수가 섞일 가능성이 낮아져 성능이 좋아진다.

Random Forest는 input과 attribute(variable), result, 이 세가지를 random하게 추출한다. 아래의 설명은 Random Forest의 종류이며, 각 bootstrap sample에서 진행된다.

- Forest-RI (random input) : Decision tree의 각 node를 분할하기 위한 기준이 되는 L개의 변수를 무작위로 추출한다. 모든 변수로 tree를 구성하지 않는다. 각각의 decision tree는 서로 다른, 무작위하게 뽑한 변수로 가지치기 없이 최대 크기로 성장시킨다.

- Forest-RC (randomly combined) : 위의 방법과 달리, 변수의 선형 결합으로 새로운 변수를 생성한다. 각 노드마다 L개의 변수가 랜덤하게 선택되고, Uniform(-1, 1)을 따르는 난수를 weight로 하여 결합한다. F개의 선형 조합이 생성하고, node를 가장 잘 분할하는 조합을 선택한다. 이 Random Forest는 단일 classifier의 correlation을 줄일 때 유용하며, 사용할 수 있는 attributes의 개수가 작을 때도 역시 유용하다.

- Decision Tree의 각 node에서 F개의 best splits 중에서 무작위로 하나를 선택하여 tree를 grow한다.

따라서 각 bootstrap sample마다 decision tree를 구성했으니, Aggregating을 진행해 회귀트리일 때 평균을, 분류트리 일 때 majority voting을 통해서 최종 결과를 도출해 낼 수 있다. 전체 변수 개수가 p개이고 선택한 변수 개수가 m개라 하면 회귀 tree는 m=p/3, 분류 tree는 m=$\sqrt{p}$로 정하는 것이 좋은 예측력을 보인다고 알려져 있다. m=p일 경우는 bagging과 같다.

변수의 임의추출을 통해서 Random Forest는 극소수의 변수들이 강한 예측력을 가지는 경우, 훈련 과정에서 여러 tree들이 그 변수들을 중복 선택하여 서로 상관화되는 것을 방지한다. 또한 변수의 선형 결합을 무작위로 진행하기에 축에 직교하게만 boundary를 형성하는 tree의 단점을 극복했다. Random Forest는 다양성을 극대화하여 예측력이 높고 다수 tree의 결과를 종합하여 안정성도 우수하다. 하지만 다수 tree를 이용해 의사 결정을 내리기에 설명력이 떨어진다.

6. Variance Importance

Bagging과 random forest는 tree 하나보다는 예측정확성이 높다. 그러나 결과 모델을 해석하기 어렵다. 변수 중요도를 설명하기 쉽다는 tree의 장점이 사라진 것이다. 많은 수의 tree를 이용했기에 어떤 변수가 예측과정에서 얼마나 중요한지 정확히 알 수 없다. 따라서 이를 해결하기 위해서는 변수 중요도를 측정하는 것이 중요하다.

각 독립변수들의 전체적인 중요도는 RSS(Residual Sum of Squares)나 Gini계수를 통해 알 수 있다. Regression tree bagging의 경우 RSS를 사용하고, classification tree bagging의 경우 Gini계수를 사용한다. 특정 인자값을 재조합하고 재조합 전후의 OOB error를 비교하는 방법(Permutation importance) 도 있다.

6.1 RSS(Residual Sum of Squares)

$$\sum_{j=1}^{J}\sum_{i\in R_j}(y_i-\hat{y}_{R_j})^2$$

R_j : X변수에 대한 J개의 겹치지 않은 영역.

$\hat{y}_{R_j}$: 각 R_j에 대한 평균 값.

RSS는 실제 y값들과 $\hat{y}_{R_j}$ 의 차이의 제곱합을 의미한다. B개의 tree에 대해 각 변수에서 split으로 인해 RSS가 많이 감소하였다면 이는 중요한 변수임을 의미한다.

6.2 Gini계수

$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk})$$

$\hat{p}_{mk}$: m영역에 속하는 레코드 중 k범주에 속하는 레코드의 비율

Gini계수가 0에 가까울수록 불균등 분배 (낮은 불순도), 1에 가까울수록 균등 분배 (높은 불순도)를 의미한다. 역시 B개의 tree에 대해 각 변수에서 split으로 인해 Gini계수가 감소된 정도를 측정하고 평균을 낸다. 해당 변수로 인해 Gini 계수가 많이 감소하였다면 이는 중요한 변수임을 의미한다.

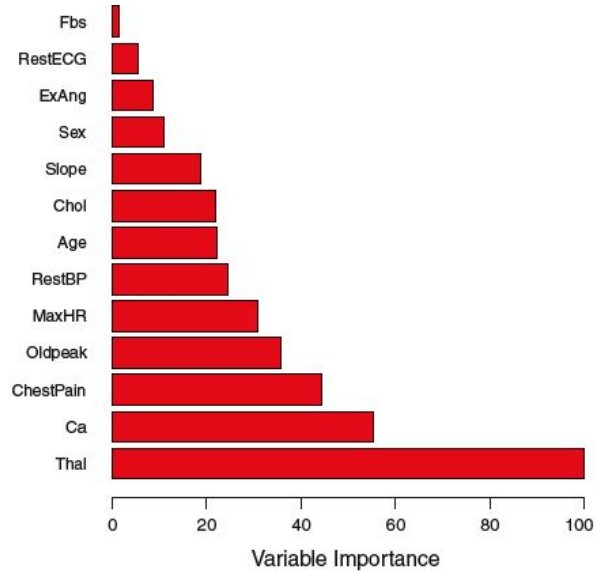
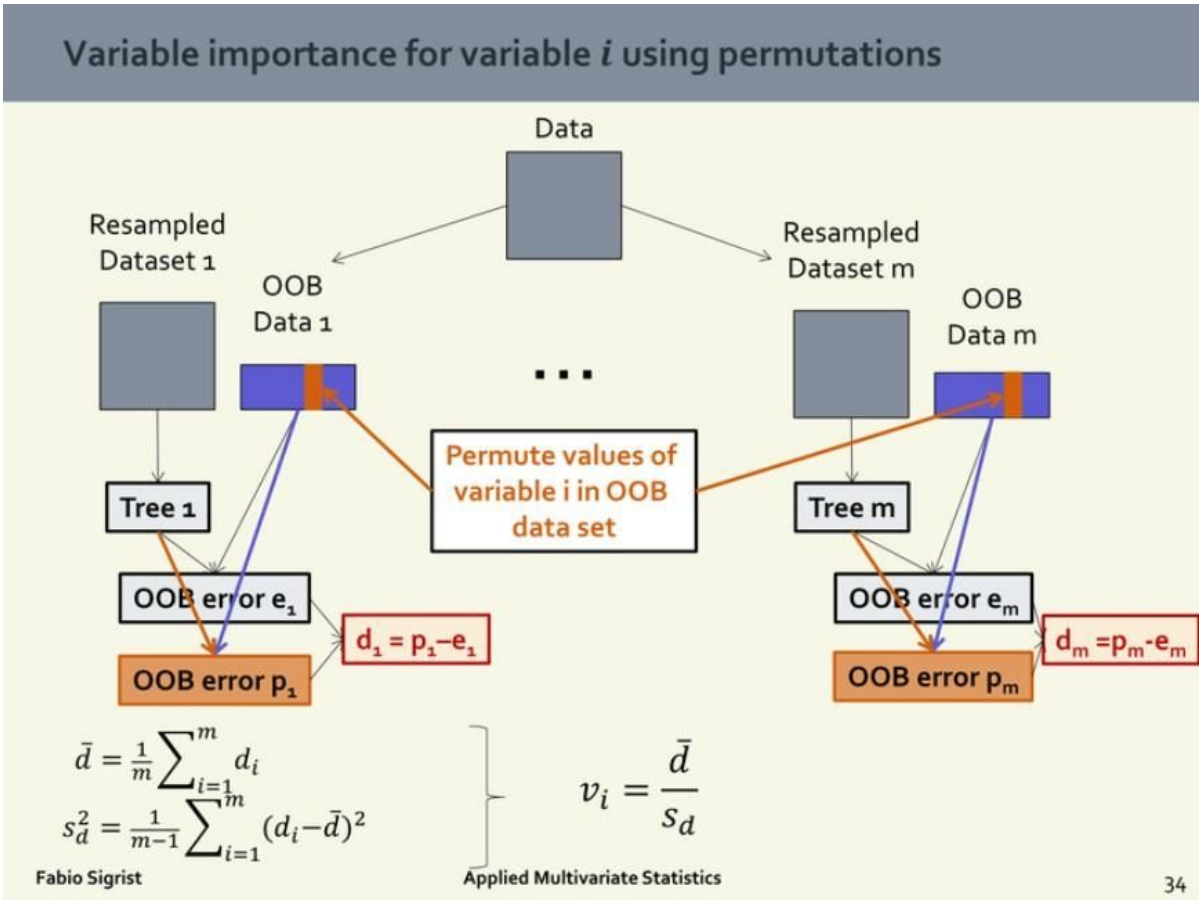


FIGURE 8.9. *A variable importance plot for the* **Heart** *data. Variable importance is computed using the mean decrease in Gini index, and expressed relative to the maximum.*

https://bit.ly/2NreLwa

6.3 Permutation importance

각 B개 tree에 대해 OOB 데이터를 이용해 오분류율(error)을 구한다. OOB 데이터의 특정 변수를 선택한 후 그 변수의 값들을 재조합(permutation)한다. 간단히 말해 변수값들의 위치를 바꾼다는 의미이다. 그 후에 OOB error를 다시 구하고 원래 OOB error와의 차이를 구한다. 만약 그 변수가 중요하지 않은 변수라면 OOB error값의 차이가 크지 않을 것이다. 이미 학습된 모델을 사용하므로 재조합 후의 데이터로 다시 학습시킬 필요가 없다. Permutation을 이용해 변수 중요도를 측정하는 방법은 간단하고, 신뢰할 수 있어서 다른 기법에서도 널리 쓰인다고 한다.



http://blog.naver.com/PostView.nhn?blogId=sw4r&logNo=221032295777

Reference

Bagging

http://euriion.com/?p=412193

https://www.kdnuggets.com/2017/11/difference-bagging-boosting.html

https://swalloow.github.io/bagging-boosting

https://bit.ly/33T6839

http://blog.naver.com/laonple/220838501228

Bootstrap

https://adnoctum.tistory.com/296

https://m.blog.naver.com/wjddudwo209/220013117867 Bootstrapping

https://bit.ly/2TW4pp9

https://en.wikipedia.org/wiki/Bootstrapping_(statistics)

Random Forest

https://bit.ly/2Pk4xAu

백준걸 교수님. 데이터 마이닝 강의안.

Random Forests, Leo Breiman, University of California Berkeley, CA 94720, January 2001, p11

Variance Importance

https://rpago.tistory.com/56

https://bit.ly/2NreLwa

http://blog.naver.com/PostView.nhn?blogId=sw4r&logNo=221032295777

https://explained.ai/rf-importance/index.html

An Introduction to Statistical Learning with Applications with R 책