알고리즘 설명자료

배정	
상태	알고리즘 설명자료
속성	

ExtraTrees Regressor (Extreme Randomized Trees)

> *class* sklearn.ensemble.ExtraTreesRegressor(*n_estimators=100*, , *criterion='squared_error'*, *max_depth=None*, *min_samples_split=2*, *min_samples_leaf=1*, *min_weight_fraction_leaf=0.0*, *max_features='auto'*, *max_leaf_nodes=None*, *min_impurity_decrease=0.0*, *bootstrap=False*, *oob_score=False*, *n_jobs=None*, *random_state=None*, *verbose=0*, *warm_start=False*, *ccp_alpha=0.0*, *max_samples=None)*

Definition

This class implements a meta estimator that fits a number of randomized decision trees (a.k.a. extra-trees) on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting.

출처: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.ExtraTreesRegressor.html

Characteristic

1. 데이터 샘플 수, 트리 형성을 위한 feature 선택까지 무작위성을 부여해 Random Forest보다 더욱 극단적으로 랜덤하게 만들어진 모델
2. Random Forest와의 차이점

   ▼ 부트스트래핑의 유무

   - Random Forest : Bagging 기법 사용, Bootstrapping을 기반으로 weak tree를 생성한 후, 각 weak tree가 서로 다른 분포의 데이터를 학습하고 이를 ensemble 하는 방식
   - Extra Trees : Bagging 기법이 아님, Bootstrapping을 하지 않고 각 결정 트리를 만들 때 **Whole Origin Data를 그대로 사용** (i.e. 비복원추출)

   ▼ Split에 쓸 Feature 선택 기준

   - Random Forest : 주어진 샘플에 대해 모든 변수(Feature)에 대한 Information Gain을 계산한 후 가장 높은 Information Gain을 가지는, 즉 설명력이 가장 높은 변수를 Split 기준으로 선택

     → 모든 변수에 대한 Information Gain을 계산한 후 비교해야 하므로 연산량이 증가
   - Extra Trees : Split을 할 때 **무작위로 Feature 선정**

     → 모든 Feature 중 무작위로 하나를 고른 후, 그 Feature에 대해서만 최적의 Partition을 찾아 노드 분할
3. 장점

- Bootstrapping을 쓰지 않고 Whole Origin Data를 사용하므로 Random Forest에 비해 Bias를 낮출 수 있음
- Split point를 랜덤하게 선택해 Variance를 줄일 수 있음
- 연산 속도 개선

Hyper Parameter Tuning

1. **n_estimators**
   - default = 100
   - 트리의 개수 결정
   - 값이 클수록 일반화된 결과 생성이 가능하지만 소요 시간이 증가
2. **max_depth**
   - default = None
   - 트리의 최대 깊이
   - 값이 증가함에 따라 초반에는 test set에 대한 성능이 증가하지만 특정 시점 이후에는 train set에 과적합이 발생해 성능 감소
3. **min_samples_split**
   - default = 2
   - 노드를 분할하는데 필요한 최소 샘플 수
   - 작게 설정할수록 분할 노드가 많아져 과적합 가능성 증가
   - 너무 크게 설정하면 분할이 잘 이루어지지 않아 성능 감소
4. **min_samples_leaf**
   - default = 1
   - leaf 노드의를 만들기 위해 필요한 최소 샘플 수
   - 값이 증가함에 따라 과적합 방지

XGBoost (Extreme Gradient Boosting)

💡 xgboost.train(*params*, *dtrain*, *num_boost_round=10*, , *evals=None*, *obj=None*, *feval=None*, *maximize=None*, *early_stopping_rounds=None*, *evals_result=None*, *verbose_eval=True*, *xgb_model=None*, *callbacks=None*, *custom_metric=None*)

Definition

XGBoost is an optimized distributed gradient boosting library designed to be highly efficient, flexible and portable. It implements machine learning algorithms under the Gradient

Boosting framework. XGBoost provides a parallel tree boosting (also known as GBDT, GBM) that solve many data science problems in a fast and accurate way.

출처: https://xgboost.readthedocs.io/en/latest/index.html

Characteristic

1. Gradient Boosting 알고리즘(여러 개의 약한 Decision Tree를 Ensemble)을 분산환경에서도 실행할 수 있도록 구현해 놓은 라이브러리 (병렬 학습 지원)
2. 장점
   - 병렬 처리로 학습 및 분류 속도가 빠름
   - 자체 과적합 규제 기능으로 강한 내구성을 지님
   - Early Stopping(조기 종료) 기능 존재
   - 다양한 옵션을 제공해 customizing 용이

LGBM (Light Gradient Boosting Machine)

💡 **lightgbm.train(*params*, *train_set*, *num_boost_round=100*, *valid_sets=None*, *valid_names=None*, *fobj=None*, *feval=None*, *init_model=None*, *feature_name='auto'*, *categorical_feature='auto'*, *early_stopping_rounds=None*, *evals_result=None*, *verbose_eval='warn'*, *keep_training_booster=False*, *callbacks=None*)**

Definition

LightGBM is a gradient boosting framework that uses tree based learning algorithms. Many boosting tools use pre-sort-based algorithms for decision tree learning. It is a simple solution, but not easy to optimize. LightGBM uses histogram-based algorithms which bucket continuous feature (attribute) values into discrete bins. This speeds up training and reduces memory usage.

출처 : https://lightgbm.readthedocs.io/en/latest/index.html

Characteristic

1. Tree구조가 수평적으로 확장하는 다른기존 Tree기반 알고리즘에 비해 수직적으로 확장을 한다. (leaf-wise 방식)
2. 장점
   - 학습하는데 시간이 적게 걸린다
   - 메모리 사용량이 상대적으로 적은 편이다
   - 카테고리형 피처들의 자동 변환과 최적 분할

- 확장을 위해 max delta loss를 가진 leaf를 선택하게 되므로 level-wise 알고리즘과 비교하여 더 많은 손실을 줄일 수 있다
- 최신의 것이기에 기능상의 다양성이 더 많다

Hyper Parameter Tuning

1. **num_iterations**
   - default = 100
   - 반복 수행하려는 트리의 개수를 지정
   - 너무 크게 설정하면 과적합 발생
2. **learning_rate**
   - default = 0.1
   - 부스팅을 반복 수행할 때 업데이트 되는 학습률
   - 0~1 사이의 값을 지정
3. **max_depth**
   - default = 1
   - 트리의 최대 깊이
   - 0보다 작은 값을 입력하면 깊이에 제한 없음
4. **min_data_in_leaf**
   - default = 20
   - 의사결정나무의 min_samples_leaf와 같은 파라미터
   - 과적합을 제어해주는 파라미터로 사용
5. **num_leaves**
   - default = 3
   - 하나의 트리가 가질 수 있는 최대 리프 개수
   - 개수를 높이면 정확도가 높아지지만 모델이 너무 복잡해지면 과적합 발생 가능성이 커진다
6. **bagging_fraction**
   - default = 1.0
   - 데이터를 샘플링하는 비율
   - 과적합을 제어하기 위해 사용
7. **feature_fraction**
   - default = 1.0
   - 개별 트리를 학습할 때 무작위로 선택하는 피처의 비율

ENSEMBLE

ExtraTrees Regressor, XGBoost , LGBM 3개의 모델을 2:5:3 비율로 앙상블