Week 7 스터디 발표

Missing data Imputation

Missing data: 결측값

데이터에서 missing이 발생한 경우, missing이 발생한 변수와 결측 비율에 따라 imputation 방법이 달라진다.

[분석 데이터 missing 비율]

```
missing = pd.Series(app_train.isna().sum().sort_values(ascending = False))
missing = missing[missing > 0]
missing = missing/len(app_train)
missing
```

```
COMMONAREA_MEDI             0.698723
COMMONAREA_AVG              0.698723
COMMONAREA_MODE             0.698723
NONLIVINGAPARTMENTS_MODE    0.694330
NONLIVINGAPARTMENTS_MEDI    0.694330
                              ...
EXT_SOURCE_2                0.002146
AMT_GOODS_PRICE             0.000904
AMT_ANNUITY                 0.000039
CNT_FAM_MEMBERS             0.000007
DAYS_LAST_PHONE_CHANGE      0.000003
```

→ 일반적으로 missing 비율이 80~90% 이상이면 해당 변수 자체를 drop 한다.

→ 하지만 missing 비율이 80% 이하이면 해당 변수를 drop 했을 때 데이터 및 정보 손실이 발생할 수 있다.

→ 따라서 적절한 imputation을 이용해 missing 데이터를 다른 값으로 대체한다.

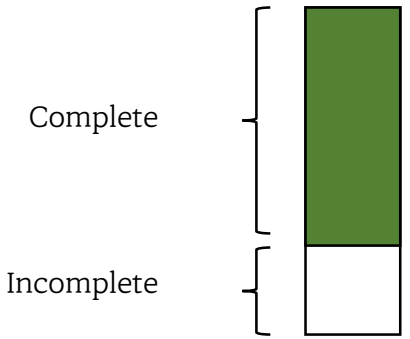
Imputation Methods

(1) Simple imputation: Mean, mode, median imputation

해당 imputation 방법들은 간단하며 연산 속도도 빠르지만, 분포의 왜곡과 분산의 과소 추정이 발생할 수 있는 위험이 있다.
또한 해당 방법은 데이터의 분포를 고려하지 않고 하나의 값으로 missing value를 impute한다는 단점이 있다.

(2) Hotdeck imputation

Missing value를 해당 변수의 랜덤하게 선택된 값으로 대체하는 방법이다. Hotdeck의 경우 (1)과 달리 분포의 왜곡이 적다. 하지만 어떤 값을 선택하여 impute 하는지에 따라 성능이 달라질 수 있다는 단점이 있다.



Alternative: Use strong algorithm!

XGBoost, LightGBM, CatBoost와 같은 Boosting 계열 머신러닝 알고리즘의 경우, 결측값을 impute하지 않아도 알아서 결측으로 인식하고 처리하기 때문에 imputation에 대한 고민을 할 필요가 없다는 장점이 있다. (성능 또한 우수하다고 알려져있음)

→ 참고한 kernel에서도 해당 방법으로 missing value를 처리함