결측 데이터의 유형과 처리

이선우

목차

01 개요

결측치의 의미와 결측치 처리에 대한 개략적인 내용을 알아보자.

01 개요

결측치(Missing feature)란?

- 일부 변수에서 관측 값이 누락되어 'NA(not available)'로 남은 것
- 데이터 셋은 다양한 이유로 결측치을 포함하며 NaN, 공백 등으로 나타남
- 결측치의 유형: 완전 무작위 결측, 무작위 결측, 비 무작위 결측

	col1	col2	col3	col4	col5
0	2	5.0	3.0	6	NaN
1	9	NaN	9.0	0	7.0
2	19	17.0	NaN	9	NaN

결측치 처리

- 결측치는 데이터를 분석할 때 방해가 되므로 적절한 방법으로 처리해주어야 함
- 결측치를 처리하는 데에는 다양한 방법이 존재
- 결측치가 있는 데이터 샘플을 삭제하는 방법도 있지만 결측치가 많을 경우 데이터 손실이 크고 편향이 발생
- 결측치의 처리 방법: 아무것도 안 하기, 평균값이나 중앙값으로 대체, 최빈값이나 0 또는 상수로 대체, k-NN 알고리즘을 이용한 대체, MICE알고리즘을 이용한 대체, 딥 러닝을 이용한 대체

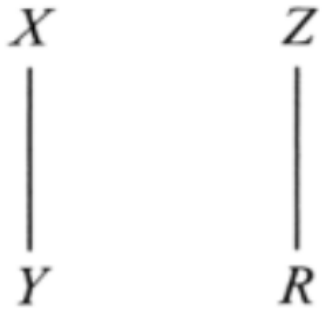
02 결측치의 유형

결측치의 3가지 유형과 예시에 대해 알아보자.

02 결측치의 유형

1. 완전 무작위 결측 (MCAR: Missing Completely At Random)

- 결측치의 발생이 다른 변수와 상관 없는 경우
- 특정 변수의 결측치가 완전히 독립적으로 발생한다고 가정
- 결측치의 존재가 문제되지 않아 이상적인 경우이지만 현실에서는 그럴 가능성이 높지 않음
- ex) 전산오류, 통신문제 등으로 인한 데이터 누락

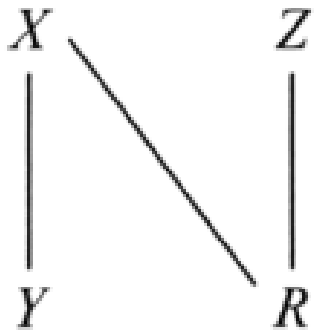


(a) MCAR

02 결측치의 유형

2. 무작위 결측 (MAR: Missing At Random)

- 결측치의 발생이 특정 변수와 관련이 있으나 얻고자 하는 결과와는 상관이 없는 경우
- 결측 발생이 관측된 값에 의해서만 설명되고 결측된 값과는 독립일 것이라고 가정
- 관측된 값으로부터 결측치를 추정하는 것이 가능하므로 다양한 결측치 대체 방법 적용 가능
- ex) 30대 남성이 용돈 설문을 할 때 결측치가 자주 발생. 30대 남성과 용돈 설문 결측에는 관련이 있으나 소득 수준(얻고자 하는 결과)과 용돈 설문과는 상관관계가 없음.

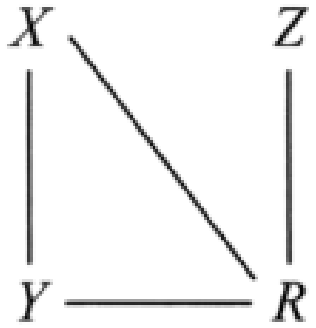


(b) MAR

02 결측치의 유형

3. 비 무작위 결측 (MNAR: Missing Not At Random)

- 결측치의 발생이 다른 변수에 의해서 결정되는 경우
- 결측치 발생이 관측된 값과 결측된 값 모두에 영향을 받는 상태
- 결측치 발생에 어떠한 이유가 있는 상태이므로 결측치에 대해 세세하게 추가 조사 필요
- ex) 용돈 설문에서 소득이 낮은 사람이 자신의 소득수준에 대해 응답하기를 꺼려함. 소득 수준과 용돈 설문의 결측에 상관성이 있음



(c) MNAR

03 결측치의 처리

결측치의 6가지 처리 방법과 장단점에 대해 알아보자.

03 결측치의 처리

1. Do Nothing

- 빈 데이터도 학습 입력 범위로 간주하거나 무시하여 분석에 사용
- 장점: 아무 것도 안 해도 되기 때문에 구현 할 필요 없음
- 단점: 일부 분석 모델에서 작동 안 함

03 결측치의 처리

2. Imputation Using (Mean/Medium) Values

- Imputation: 결측치를 특정 값으로 대신 채우는 결측치 처리 방법
- 각 컬럼에서 정상값들의 평균값, 중앙값을 계산한 후 해당 컬럼의 결측치를 대체
- 평균값(Mean): 데이터를 모두 더한 후 데이터의 개수로 나눈 값
- 중앙값(Medium): 전체 데이터 중 가운데에 있는 수
  데이터의 개수가 짝수인 경우는 가운데 두 수의 평균값

	col1	col2	col3	col4	col5
0	2	5.0	3.0	6	NaN
1	9	NaN	9.0	0	7.0
2	19	17.0	NaN	9	NaN

	col1	col2	col3	col4	col5
0	2.0	5.0	3.0	6.0	7.0
1	9.0	11.0	9.0	0.0	7.0
2	19.0	17.0	6.0	9.0	7.0

03 결측치의 처리

2. Imputation Using (Mean/Medium) Values

- 장점
  - 쉽고 빠름
  - 데이터 샘플이 많지 않은 데이터 셋에서는 성능이 우수함
- 단점
  - 단순히 결측치가 존재하지 않는 컬럼만 고려하기 때문에 다른 변수 간의 상관관계를 고려하지 않음
  - 컬럼 레벨에서만 적용할 수 있음
  - 숫자값을 갖는 데이터에만 적용할 수 있음
  - 범주형 변수(categorical features)에는 부적합 →
  - 정확도가 떨어짐

(1) 범주형 변수: 성별이나 종교와 같이 고유한 값이나 범주 수가 제한된 변수
(2) 연속형 변수: 온도, 키, 체중과 같이 연속적인 수로 수량화 가능한 변수

03 결측치의 처리

2. Imputation Using (Mean/Medium) Values

- **SimpleImputer**를 이용하여 결측치가 포함된 컬럼의 평균값/중앙값으로 대체 가능

```
# SimpleImputer, strategy를 'mean'으로 지정하면 결측치들을 해당 column의 평균값으로 대치한다.
# strategy에 'mean':평균,'median':중간값,'most_frequent':최빈값 등 다양한 방법을 적용하여 값을 대치할 수 있다.
imputer = SimpleImputer(missing_values=np.nan, strategy="mean", add_indicator=True)
```

03 결측치의 처리

3. Imputation Using (Most Frequent) or (Zero/Constant) Values

- 최빈값이나 0 또는 상수로 결측치를 대체
- 최빈값: 데이터 집합에서 가장 빈번하게 등장한 데이터

	0	1	2	3	4
0	NaN	41.0	6.984127	1.023810	322.0
1	8.3014	21.0	6.238137	0.971880	2401.0
2	NaN	52.0	8.288136	1.073446	496.0
3	5.6431	NaN	5.817352	NaN	558.0
4	NaN	52.0	6.281853	1.081081	565.0
5	4.0368	NaN	4.761658	1.103627	413.0
6	3.6591	52.0	4.931907	0.951362	1094.0
7	NaN	52.0	4.797527	1.061824	1157.0
8	NaN	42.0	4.294118	1.117647	1206.0
9	3.6912	52.0	4.970588	0.990196	1551.0

	0	1	2	3	4
0	3.6591	41.0	6.984127	1.023810	322.0
1	8.3014	21.0	6.238137	0.971880	2401.0
2	3.6591	52.0	8.288136	1.073446	496.0
3	5.6431	52.0	5.817352	0.951362	558.0
4	3.6591	52.0	6.281853	1.081081	565.0
5	4.0368	52.0	4.761658	1.103627	413.0
6	3.6591	52.0	4.931907	0.951362	1094.0
7	3.6591	52.0	4.797527	1.061824	1157.0
8	3.6591	42.0	4.294118	1.117647	1206.0
9	3.6912	52.0	4.970588	0.990196	1551.0

03 결측치의 처리

3. Imputation Using (Most Frequent) or (Zero/Constant) Values

- 장점
  - 쉽고 빠름
  - 문자열이나 숫자로 표현된 범주형 변수에 대해서도 사용 가능
- 단점
  - 평균/중앙값 대체와 마찬가지로 변수 간 상관관계를 고려하지 않음
  - 데이터에 편향성을 만들어 낼 수 있음

03 결측치의 처리

3. Imputation Using (Most Frequent) or (Zero/Constant) Values

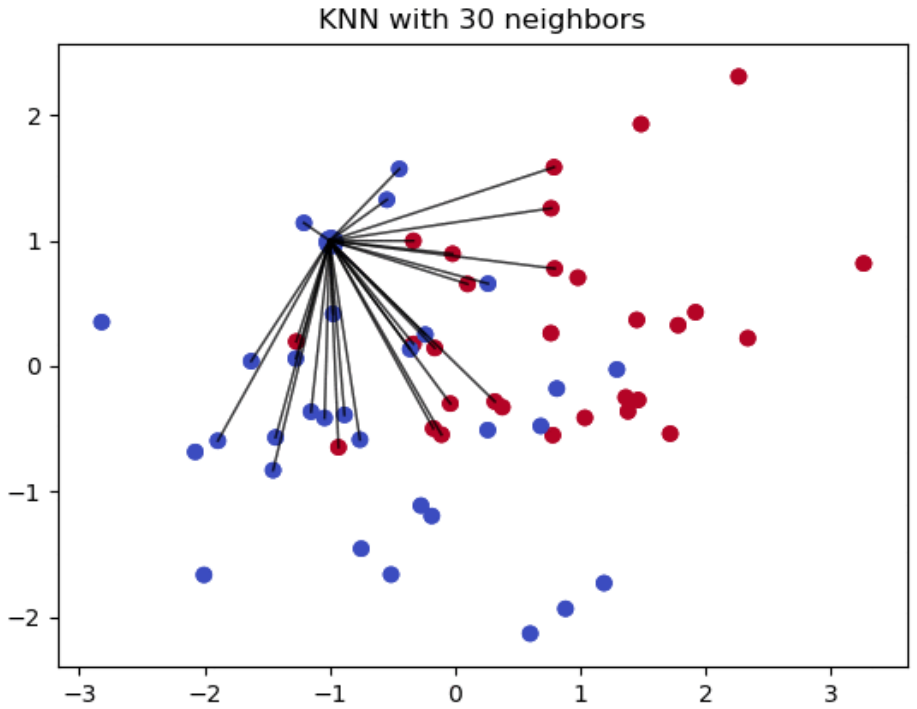
- **SimpleImputer**를 이용하여 결측치를 최빈값이나 0또는 상수로 채움
-

```
# SimpleImputer, strategy를 'constant'로 지정하고 fill_value를 '0'으로 지정하면 결측치를 0으로 채운다.
# fill_value에 원하는 상수를 입력하여 해당 값으로 결측치를 대치할 수 있다.(string or numeric data)
imputer = SimpleImputer(
    missing_values=np.nan, add_indicator=True, strategy="constant", fill_value=0
)
```

03 결측치의 처리

4. Imputation Using k-NN

- k-NN(k-nearest neighbor) 회귀 알고리즘을 이용하여 분석 대상을 중심으로 가장 가까운 k개의 이웃 데이터를 찾고 이웃 데이터들의 평균값으로 결측치를 대체



03 결측치의 처리

4. Imputation Using k-NN

- 장점
  - 일반적으로 중앙값, 평균값 대체나 최빈값 대체보다 정확한 결과를 얻을 수 있음
- 단점
  - 계산량이 많고 메모리가 많이 필요
  - 적절한 k의 선택이 필요
  - 이상치(outlier)에 민감
  - 고차원 데이터에서 부정확할 수 있음

03 결측치의 처리

4. Imputation Using k-NN

- **KNNImputer**를 이용하여 결측치 대치 진행 (k-NN 알고리즘 사용)
- 각 샘플의 결측값은 train set에서 가장 가까운 k개 이웃의 평균값을 사용하여 대체 가능
-

```
# n_neighbors 파라미터를 통해서 가장 가까운 이웃의 수를 정할 수 있다.(현재 default값인 5로 지정되어 있음)
imputer = KNNImputer(missing_values=np.nan, add_indicator=True)
```

03 결측치의 처리

5. Imputation Using Multivariate Imputation by Chained Equation (MICE)

- 다변량 대치법(MI) 중 하나인 다변량 연쇄 방정식(MICE)을 이용하여 결측치를 대체
- 단변량 대치법(Single Imputation)
  - 평균 대치법과 같이 대치법을 한 번 수행하여 결측치를 채우는 방법
- 다변량 대치법(Multiple Imputation, MI)
  - 단순하게 한 번 대체를 수행하는 것 보다 여러 대체를 조합하는 것이 더 좋다는 개념
  - 1) Imputation: 단변량 대치법을 거친 여러 개의 데이터 셋을 생성

    2) Analysis: 완성된 여러 개의 데이터 셋을 분석

    3) Pooling: 평균, 분산, 신뢰 구간을 계산하여 결과를 하나로 합침

03 결측치의 처리

5. Imputation Using Multivariate Imputation by Chained Equation (MICE)

- 다변량 연쇄 방정식(MICE): 첫 변수 x1은 다른 모든 변수 x2 ~ xn에 대해 회귀함(x1의 결측치는 사후 분포에서 시뮬레이션된 값으로 대체). 다음 변수인 x2는 x3 ~ xn에 대해 회귀함. 결측치는 사후 분포에서 추출된 값으로 대체되고, 결측치의 개수만큼 연쇄적인 특징을 가짐

0	2	5.0	3.0	6.0	NaN
1	9	NaN	9.0	0.0	7.0
2	19	17.0	NaN	9.0	NaN
3	7	10.0	3.0	6.0	4.0
4	2	8.0	10.0	NaN	3.0

0	2.0	5.0	3.00	6.00	4.666667
1	9.0	10.0	9.00	0.00	7.000000
2	19.0	17.0	6.25	9.00	4.666667
3	7.0	10.0	3.00	6.00	4.000000
4	2.0	8.0	10.00	5.25	3.000000

03 결측치의 처리

5. Imputation Using Multivariate Imputation by Chained Equation (MICE)

- 장점
  - 대치로 인한 노이즈 증가 문제 해결
  - 단변량 대치법보다 성능이 좋음
  - 복잡도가 높은 데이터에서 좋은 결과를 가져옴
- 단점
  - 대용량 데이터일수록 연산량이 많아져 속도가 느림

03 결측치의 처리

5. Imputation Using Multivariate Imputation by Chained Equation (MICE)

- **IterativeImputer**를 이용하여 결측치 대치 진행 (MICE 알고리즘 사용)
-
```
# sample_posterior=True로 지정하면 Multiple Imputation과 같은 동작을 수행한다.
imputer = IterativeImputer(
    missing_values=np.nan,
    add_indicator=True,
    random_state=0,
    n_nearest_features=3,
    max_iter=1,
    sample_posterior=True,
)
```

03 결측치의 처리

6. Imputation Using Deep Learning (Datawig)

- 결측치가 있는 변수에 대해 딥 러닝을 적용하여 모델을 학습시켜 결측치를 대체
- Datawig: 심층 신경망(DNN)을 사용해서 데이터 셋에 존재하는 결측값을 채우도록 머신러닝 모델을 훈련시키는 라이브러리
- 장점
  - 카테고리 변수와 숫자값이 아닌 변수도 학습 가능
  - 다른 대치법에 비해 성능이 매우 좋고 정확도가 높음
  - CPU, GPU를 지원함
- 단점
  - 대용량 데이터에서는 속도가 느림
  - 결측치를 채워 넣을 타겟 컬럼과 상관성이 높거나, 타겟 컬럼의 정보를 포함하고 있는 다른 컬럼들을 직접 지정해주어야 함

03 결측치의 처리

6. Imputation Using Deep Learning (Datawig)

- **Imputer**를 선언하고 학습, 예측하여 결측치 대치 진행(Datawig 라이브러리 사용)

```
# imputer 선언
imputer = datawig.SimpleImputer(
    input_columns=['year', 'month'], # 결측값 예측 모델 학습에 사용할 column들
    output_column='passengers' # 결측값을 예측하고자 하는 column
    )

# Training
imputer.fit(train_df = flights_train,num_epochs=200,batch_size=32)

# Prediction
imputed = imputer.predict(flights_test)
```