2023 D&A

ML Session 4차시

Data Preprocessing

	수1							목3						
김지은	김소현	민수홍	송은아	신지후	조현식		이서연	김상욱	박준하	배성윤	이지민	이지안	
	수2							목4						
천예은	송승원	이유정	이재원	이호영	임하영	황예은	김현조	권민지	김차미	김훈래	편서연		
	목1							금1						
이현준	김윤재	박세현	백경린	홍예진			김채원	김세은	김하경	안병민	윤청현	이서진	최준일
멘토	목2													
신기섭	김다희	김진하	손아현	신유인	이준혁	정용재							

CONTENTS

Review
지난 시간 요약

회귀 모델

선형/다항 회귀, 규제 (Ridge, Lasso + a), Logistic Regression

분류 모델

SVM(선형/비선형), KNN, Decision Tree

Overview

ML Process

Domain Understanding & Data Collection

Data Preprocessing

Modeling & Ensemble

Evaluation

Deployment

- Data Cleansing
- Scaling/Encoding
- Feature Extraction
- Feature Selection

Data Preprocessing
데이터 전처리

Data Preprocessing이란

데이터를 가공하고 처리하는 과정으로서, 머신러닝에서 가장 많은 시간과 노력을 투자해야 하는 단계

결측치, 이상치를 처리하고 feature을 만듦

많은 feature을 만들고 유의미하다고 판단되는 feature을 feature selection을 통해 골라서 사용

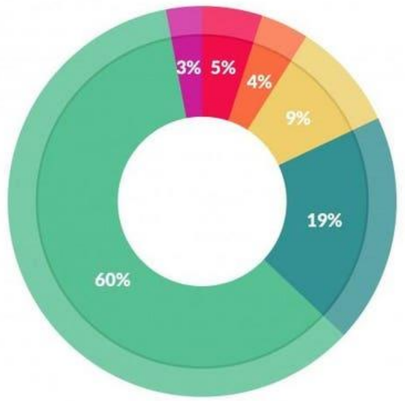
모델이 값을 잘 예측할 수 있는 유의미한 feature을 제공해야 성능이 좋은 모델을 만들 수 있음

Data Preprocessing

Data Cleansing

Data Cleansing의 정의 및 목적

데이터 분석 전, 데이터를 분석에 적합한 형태로 처리하는 과정
동일한 머신러닝 기법을 적용해도 전처리에 따라 다른 결과가 나옴



Forbes에서 인용한 CrowdFlower의 설문 결과에 따르면,
데이터 분석가는 업무 시간 중 60% 정도를 데이터 전처리 과정에 사용

Data Cleansing 과정

1. 결측치 처리 2. 이상치 처리 3. Scaling 4. Encoding

Data Cleansing
결측값 처리

결측값 처리의 사용

결측값이 있는 상태로 모델을 만들 경우, 변수 간의 관계가 왜곡될 수 있음

결측값이 발생하는 유형, 혹은 해당 Feature의 특성에 따라 결측값을 올바르게 처리해야 함

결측값 처리의 종류

1. 삭제 2. 대체

Data Cleansing
결측값 처리 - 삭제

결측값 삭제

주로 결측값이 완전 무작위로 발생한 경우에 사용
Ex) 결측값이 변수의 성격과 전혀 무관하게 발생한 경우
반면, 무작위 결측치가 아닐 경우에 결측치를 삭제해 버리면 왜곡된 모델이 생성될 수 있음

삭제 방법

전체 삭제 – 결측값이 발생한 모든 관측치를 삭제
→ 간편한 반면, 관측치가 줄어들어 모델의 유효성이 낮아짐

부분 삭제 – 모델에 포함시킬 변수들 중 결측값이 발생한 모든 관측치 삭제
→ 모델에 따라 변수가 제각각 다르기 때문에 관리 Cost가 늘어남

*보통 결측치가 50% 이상일 경우, 해당 변수를 제거함

Data Cleansing
결측값 처리 – 대체(1)

한가지 값으로 대체하는 경우

1) 평균값 대체: Column 내 값들의 평균으로 결측치 대체
→ 연속형 변수만 사용 가능

2) 중앙값 대체: Column 내 값들의 중앙값으로 결측치 대체
→ 연속형 변수만 사용 가능

3) 최빈값 대체: Column 내 값들 중 가장 많이 나온 값으로 결측치 대체
→ 연속형, 범주형 모두 사용 가능

→ *Imputer을 사용하기도 함

*Imputation: missing 데이터가 있을 때, 이것을 대체 될 수 있는 값들로 대체 하는 방법
이는 Single Imputation(특정값 대체)와 Multiple Imputation(다중값 대체)으로 나뉨

Data Cleansing
결측값 처리 – 대체(2)

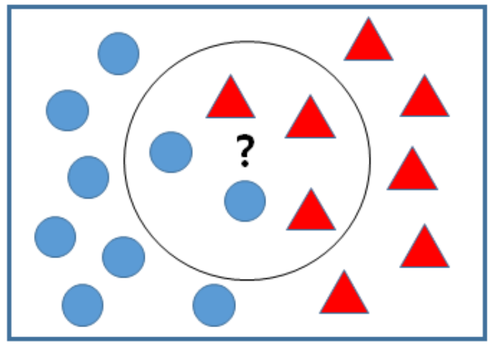
여러 가지 값으로 대체하는 경우

결측치가 아닌 데이터들을 train으로 두고, model을 돌려 값을 예측

1) KNN Imputation: KDTree를 구성한 후, 최근접 이웃을 계산해 K-NN을 찾은 후 가중 평균을 취하는 방식
쉽게 말하면, 원하는 이웃 수(n_neighbors)의 평균값을 사용해서 대치하는 방법
문자열은 들어갈 수 없음

2) Mice Imputation: Multiveriate Imputation by Chained Equations
연쇄 방정식을 이용한 대치 방법
누락된 데이터를 여러 번 채우는 방식으로 작동

k = 5 일 때 "?"는 세모로 분류됨.

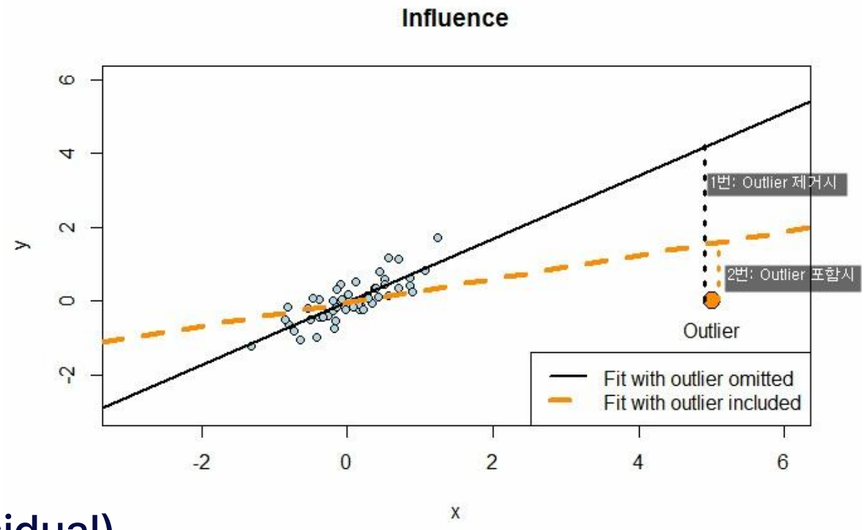


*Imputation: missing 데이터가 있을 때, 이것을 대체 될 수 있는 값들로 대체 하는 방법
이는 Single Imputation(특정값 대체)와 Multiple Imputation(다중값 대체)으로 나뉨

Data Cleansing
이상치 처리

이상치 처리의 정의

값의 범위가 일반적인 범위를 벗어나 특별한 값을 가지는 것
회귀모형*의 경우, 이상치 값에 민감하게 반응



이상치 확인

1. 시각화 : Boxplot, Histogram, Scatter plot
2. 두 변수 간 회귀 모형 : Residual, Studentized residual(or standardized residual), leverage, Cook's D 값 확인

이상치 기준 및 처리 방식

- 표준점수로 변환
- IQR 방식
- 도메인 지식 이용이나 Binning(구간화) 처리 방식

* 직선이 그어지는 위치가 이상치의 영향을 받아 이상한 곳으로 그어지는 현상으로 예시를 들 수 있음

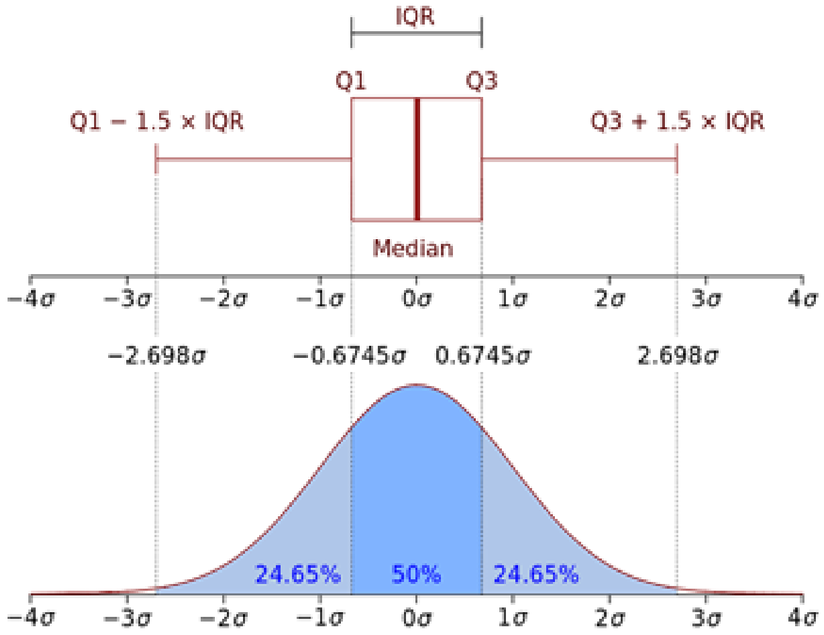
Data Cleansing
이상치 처리

표준점수로 변환

표준정규분포로 변환 후, -3 이하 3 이상의 값들을 이상치로 판단 후 제거하거나 대체하는 방식

IQR 방식

1사분위수보다 낮은 IQR의 1.5배를 벗어나는 포인트 or
3사분위수보다 높은 IQR의 1.5배를 벗어나는 포인트의 경우,
이상치로 처리함 (제거 or 대체)



Data Cleansing
총정리

결측치 처리

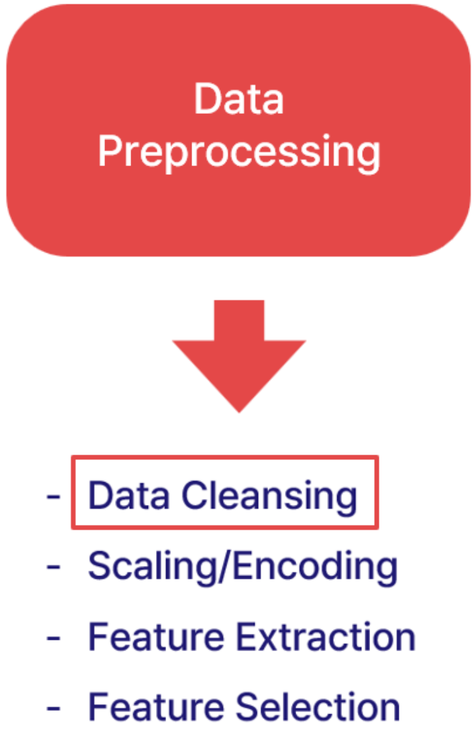
1) 삭제
- 전체 삭제, 부분 삭제

2) 대체
- 한 가지의 값으로 대체: 평균값, 중앙값, 최빈값
- 여러 가지의 값으로 대체: KNN Imputation, Mice Imputation

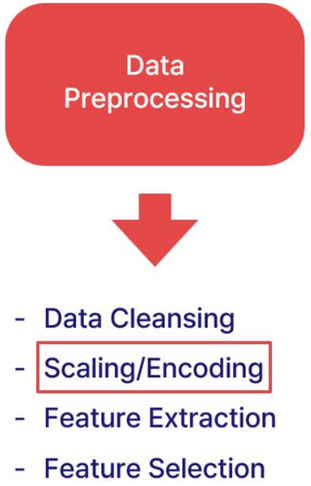
이상치 처리

1) 표준점수로 변환

2) IQR방식



Data Cleansing
Scaling



Scaling의 정의 및 목적

변수의 단위를 변경하고 싶거나,
변수의 분포가 편향되어 있을 경우,
변수 간의 관계가 잘 드러나지 않는 경우

→ 위와 같은 경우에 Scaling 수행

Scaling 방법

1. Standard Scaler 2. MinMaxScaler 3. Robust Scaler 4. Normalizer

Data Cleansing

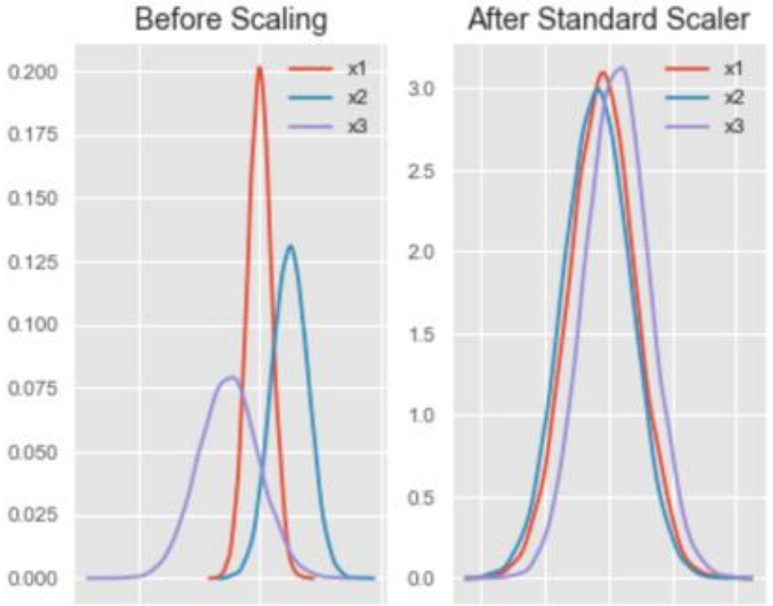
Scaling – Standard Scaler

Standard Scaler

개별 Feature에서 평균값을 빼고 분산으로 나누어 평균은 0, 분산은 1로 변환 – Standardization(표준화)
가우시안 정규 분포를 갖도록 변환하는 것은 몇몇 알고리즘에서 *매우 중요
Ex) SVM, Linear Regression, Logistic Regression, Deep Learning
하지만, 각 Feature들 사이의 상대적 거리를 왜곡시킬 수 있다는 단점 존재

* 데이터 포인트들 간의 거리를 활용해서 결정 경계를 정하는 알고리즘의 경우, 데이터 포인트에 영향을 직접적으로 받기에 더 중요

$$Y = \frac{(X - X_{mean})}{\sigma_Y}$$



Data Cleansing

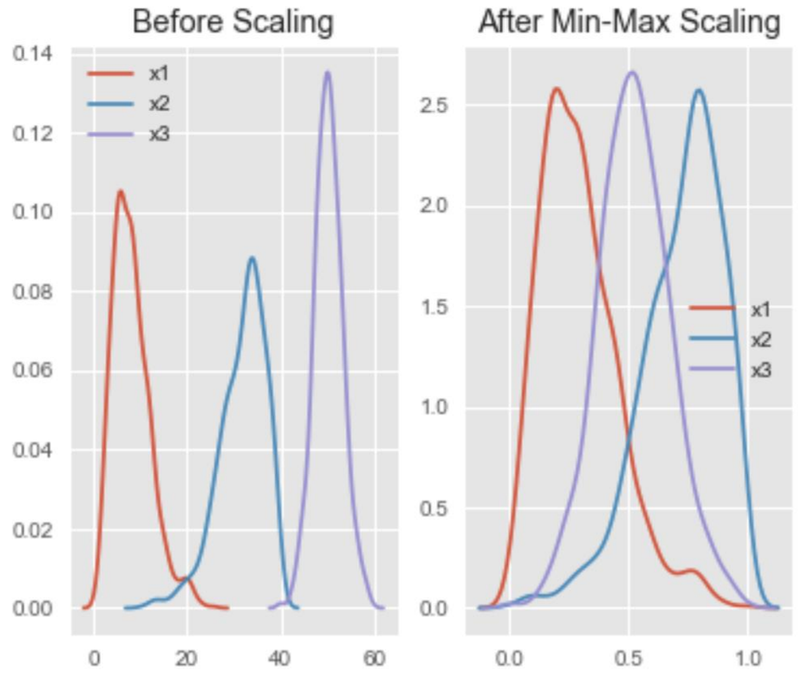
Scaling – MinMax Scaler

MinMax Scaler

개별 Feature의 크기를 모두 똑같은 단위(0~1사이)로 변경하는 것
본래 데이터의 정보를 변형시키지 않는다는 장점이 존재
반면, *이상치에 영향을 많이 받는다는 단점 존재

* 최솟값과 최댓값을 활용한 Scaler
Ex) 데이터의 대부분이 0 ~ 10 사이의 값이지만, 하나의 이상치가 100의 값을 갖는다면, 대부분의 경우, 0 ~ 0.01 값으로 변환되지만, 이상치의 경우 1로 변환하게 됨

$$Y = \frac{(X - X_{min})}{(X_{max} - X_{min})}$$



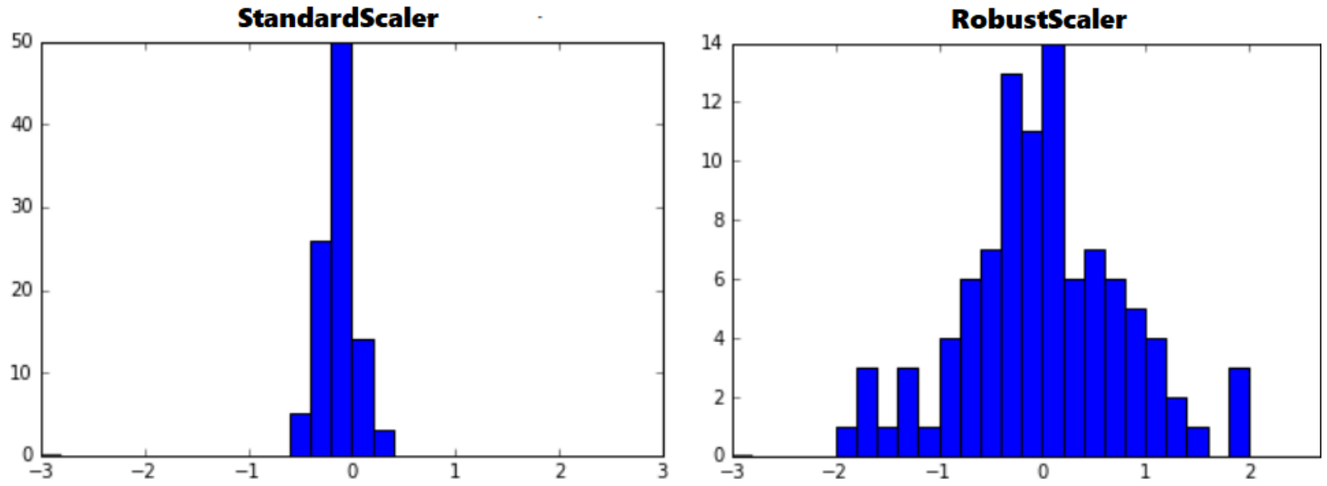
Data Cleansing

Scaling – Robust Scaler

Robust Scaler

개별 Feature 값에서 median을 빼고 IQR 범위로 나눈 것
각 Feature의 범위는 *Min-Max 보다는 큼
상대적으로 이상치에 덜 민감하게 작동
*median(값)(↔ mean)과 IQR(분포)(75%-25%=50%)은 이상치의 영향을 상대적으로 덜 받기 때문

$$Y = \frac{(X - X_{median})}{(X_{IQR,75\%} - X_{IQR,25\%})}$$



Data Cleansing

Scaling – Normalizer

Normalizer

선형대수에서의 정규화 개념이 차용되어 일반적 정규화와는 약간의 차이 존재
각 Feature의 열(column)값이 아닌 행(row)값에 적용되는 scaler
대부분의 경우, 이전에 언급된 것들이 주로 사용됨

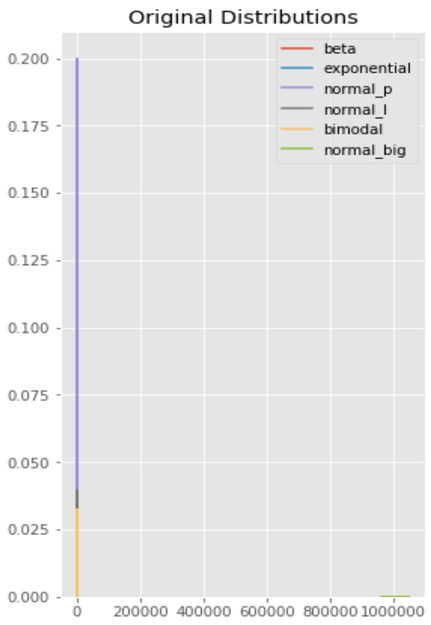
$$Y = \frac{X_i}{\sqrt{(\sum_{j=1}^{N} X_j^2)}}$$

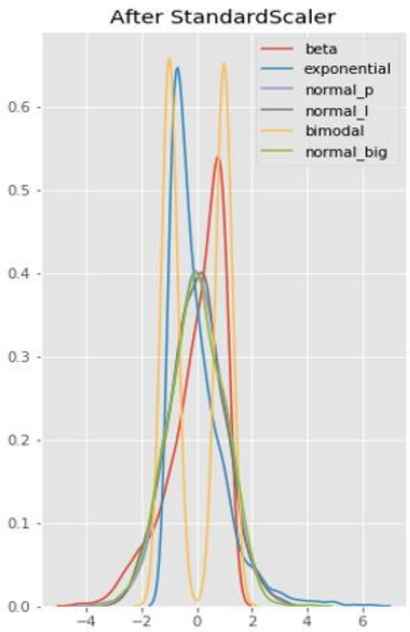
Data Cleansing

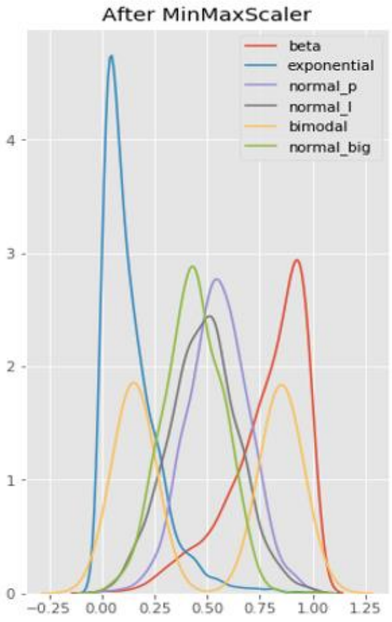
Scaling 총정리

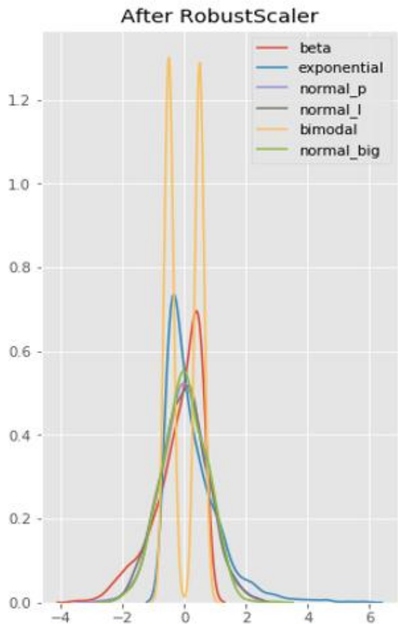
Scaler

Standard – 모든 데이터의 분포를 정규분포로 보고싶을 때
MinMax – 데이터의 왜곡없이 순수하게 분포를 비교하고자 할 때
Robust – 이상치가 존재하고 그 영향을 줄이고 싶을 때
+ Normalizer





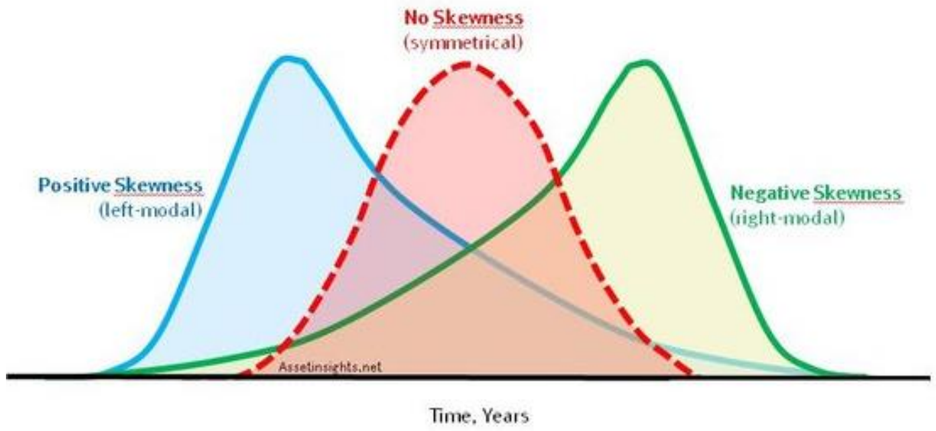




Data Cleansing
추가) Transformation

Skewness(왜도)

분포의 정규분포에 비해서 얼마나 비대칭인지 나타내는 척도
왜도 값이 양의 값 – 정규 분포보다 오른쪽에 위치
왜도 값이 음의 값 – 정규 분포보다 왼쪽에 위치



변환하는 이유?

꼬리에 있는 값을 모델에 제대로 학습시키기 위함
꼬리 부분이 상대적으로 데이터의 양이 적기 때문에 모델 학습에 반영이 적게 됨

```
if 꼬리부분 데이터 == 중요:
        if  변환하지 않은 데이터의 사용:
            print('성능 개똥') #중요한 예측을 제대로 수행하지 못함
```

Data Cleansing
추가) Transformation

Transformation의 종류

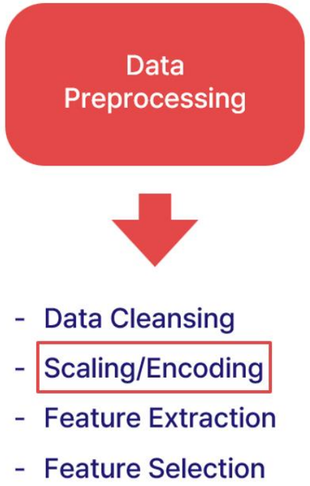
1) np.log – 로그 변환
2) np.exp – 지수 변환
3) np.sqrt – 루트 변환

*Transformation과 Scaling을 적절하게 섞어서 사용

Scaling vs Transformation

Scaling : 데이터 분포의 모양은 그대로 유지, 범위를 조정
Transformation: 데이터 분포의 모양을 변경, 분포를 조정

Data Cleansing
Encoding



Encoding의 정의 및 목적

데이터를 컴퓨터가 이해하고 처리할 수 있는 형태로 변환하는 과정
Encoding = 코드화 = 암호화
Encoding in 컴퓨터 : 컴퓨터는 문자를 이해하지 못함
데이터를 약속된 규칙에 따라 컴퓨터가 이해할 수 있는 숫자로 변환

주로 범주형 변수를 수치형 변수로 변환

Encoding의 종류

- Label Encoding
- One-hot Encoding
- Target Encoding(=Mean Encoding)

Data Cleansing

Encoding

Label Encoding

카테고리 Feature를 숫자로 변환 (ex. Apple → 1, Chicken → 2 ...)
모델은 숫자를 기반으로 연산 → *서열 변수가 아닌 경우 치명적임

1. 서로 가까운 숫자를 비슷한 데이터로 인식

Ex) 0(딸기)와 2(사과)보다 0(딸기)와 1(바나나)를 더 비슷한 데이터로 인식하는 경우 → 실제로 네 과일 모두 별개의 데이터임

2. 숫자값을 가중치로 잘못 인식

Ex) 0<1<2<3 → 딸기<바나나<사과<포도

```
encoder = LabelEncoder()
labels = encoder.fit_transform(fruit)
df['label']=labels
df
```

	과일
0	바나나
1	사과
2	사과
3	포도
4	딸기

```
sorted(set(df['과일']))
```

```
['딸기', '바나나', '사과', '포도']
```

0 1 2 3

	과일	label
0	바나나	1
1	사과	2
2	사과	2
3	포도	3
4	딸기	0
5	포도	3
6	바나나	1

*순위, 석차, 등급 등이 있음

Data Cleansing

Encoding

One-hot Encoding

Feature의 고윳값에 해당하는 Column에만 1, 나머지는 0으로 표현
차원의 저주에 걸릴 수 있음 (*sparse하기 때문)

*Ex) 어떤 Column 내의 값이 100가지라고 가정, one-hot encoding을 진행했을 때 100가지의 column이 생성
→ 데이터프레임에 0으로 채워지는 부분이 많아짐(1이 굉장히 희소해짐을 의미=데이터를 포함하는 부분이 적어짐)

Numerical value	Animal
1.5	cat
3.6	cat
42	dog
7.1	crocodile

One hot encoding →

Numerical value	Cat	Dog	Tiger	Crocodile
1.5	1	0	0	0
3.6	1	0	0	0
42	0	1	0	0
7.1	0	0	0	1

Data Cleansing

Encoding

Target Encoding (=Mean Encoding)

Label Encoding과 유사하지만, Target값과 Encoding 값이 연관이 있다는 점에서 차이가 존재
각 카테고리의 값을 학습 데이터의 Target값의 평균값으로 설정하는 방법

```
Sex
female    0.742038
male      0.188908
Name: Survived, dtype: float64
```

	Sex	Sex_mean
0	male	0.188908
1	female	0.742038
2	female	0.742038
3	female	0.742038
4	male	0.188908

Data Cleansing

Encoding

Target Encoding의 장점

카테고리의 개수가 많을수록, Label Encoding은 Label 수가 계속 증가
Target Encoding은 보다 적은 split이 생기고 학습이 더욱 빠르게 이루어짐
Encoding된 Label값이 Target과 관련된 의미를 가짐 → less bias

Target Encoding의 단점

Overfitting 가능성 높음
구현과 검증이 까다로움(*Data leakage)

*정보의 누설을 의미

Data Cleansing

Encoding 총정리

Encoding이란

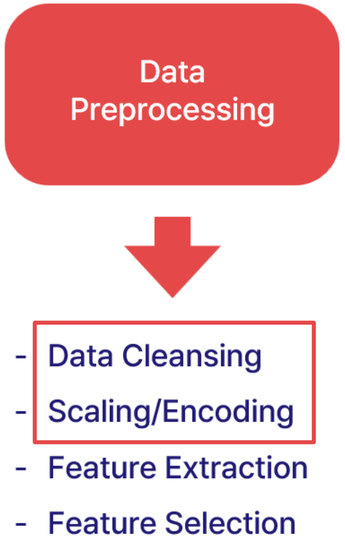
데이터를 컴퓨터가 이해하고 처리할 수 있는 형태로 변환하는 과정

Encoding의 종류

- Label Encoding
- One-hot Encoding
- Target Encoding(= Mean Encoding)

Review

Data Cleansing 총정리



결측치 처리, 이상치 처리

1) 삭제
- 전체 삭제, 부분 삭제

2) 대체
- 한 가지의 값으로 대체: 평균값, 중앙값, 최빈값
- 두 가지의 값으로 대체: KNN Imputation, Mice Imputation

1) 표준점수로 변환

2) IQR방식

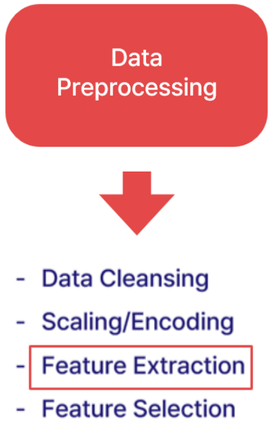
Scaling, Encoding

1) Standard – 모든 데이터의 분포를 정규분포로 보고싶을 때
2) MinMax – 데이터의 왜곡없이 순수하게 분포를 비교하고자 할 때
3) Robust – 이상치가 존재하고 그 영향을 줄이고 싶을 때
+ Transformation

1) Label Encoding
2) One-hot Encoding
3) Target Encoding(=Mean Encoding)

Overview

Feature Extraction/Selection



Feature Extraction과 Selection의 사용 이유

- 데이터의 특성 중, 모델에 중요한 정보를 제공하지 않거나 중복되는 특성들이 존재
- 따라서 중요한 Feature을 선택 or 기존 Feature의 특징 추출 등 차원을 축소하여 사용
- 차원의 저주

Feature Extraction과 Selection의 이점

- 모델의 속도 증진
- 과적합 위험 감소(차원 감소 → 모델의 복잡도 감소)
- 모델의 간결함

Feature Extraction/Selection

차원의 저주

차원의 저주 정의

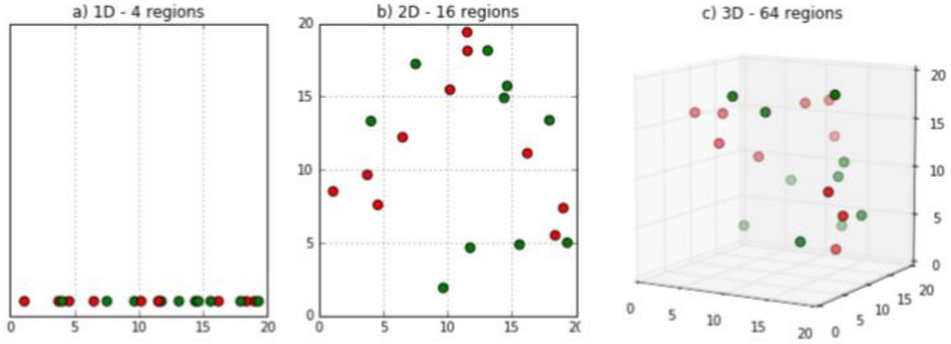
데이터의 차원이 증가할수록 필요한 데이터 양이 기하급수적으로 늘어나는 현상
→ 차원이 증가하면서 학습데이터 수가 차원 수보다 적어져 성능이 저하됨
→ 관측치보다 변수 수가 많아지는 경우, 차원의 저주 문제가 발생 (변수가 증가한다고 반드시 차원의 저주가 발생하는 것은 아님)

차원의 저주 특징

1. 데이터의 양(행)은 동일한데, 데이터의 차원(열)이 커지면 데이터의 밀도가 떨어짐
= 차원이 커질수록 데이터 간 거리가 멀어짐
→ 빈 공간이 많이 생기게 되는데, 이는 정보가 없는 공간이라고 할 수 있음
→ 따라서, 빈 공간이 많은 데이터에 대해 학습을 하게 되면 모델 성능이 저하될 수 밖에 없음
2. 원하는 정보를 찾는 데에 Computing Cost가 많이 소요

→ 데이터의 차원을 낮춰서 학습을 진행
(데이터 차원을 낮추는 방법: Extraction, Selection)

(빈 공간은 0으로 채워진 공간)



Feature Extraction/Selection

두 기법의 차이

Feature Extraction

기존 Feature에 기반하여 새로운 Feature 생성
Ex) PCA

Feature Selection

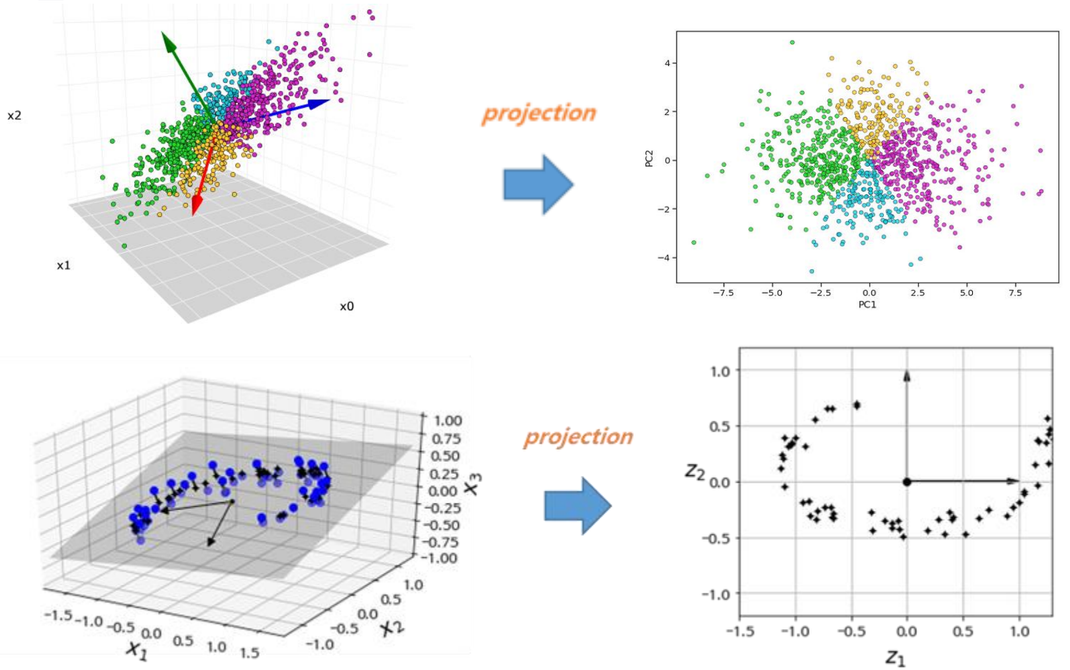
기존 Feature들의 부분 집합으로
일부 중요한 Feature들만 선택적으로 사용
Ex) Shap, Lime

Feature Extraction

PCA (Principal Component Analysis)

PCA의 정의

주성분 분석이라고도 말함
고차원의 데이터를 저차원의 데이터로 축소시키는 방법 중 하나
훈련 데이터에 가장 가까운 초평면(hyperplane)을 정의한 다음, 그 평면에 투영하는 기법
분산이 최대로 보존되는 축을 선택하는 것이 정보가 가장 적게 손실되므로 중요함



PCA의 장점

1. 시각화
   3차원이 넘어간 시각화는 우리 눈으로 볼 수 없기 때문에
   PCA를 통해 차원을 축소하여 시각화 → 데이터 패턴을 쉽게 인지 가능
2. 노이즈 제거 – 쓸모없는 Feature을 제거함으로써 노이즈 제거 가능
3. 메모리 절약
4. 퍼포먼스 향상 – 불필요한 Feature을 제거해 모델 성능 향상에 기여

Feature Selection Preview

Feature Selection의 정의

Feature Selection의 정의

기존 Feature에서 원하는 Feature만 선택하는 방법

Feature Selection의 장점

- 사용자가 해석하기 쉽게 모델을 단순화
- 훈련 시간 축소
- 차원의 저주 방지
- 일반화

Review
총정리

Feature Extraction

기존 Feature에 기반하여 새로운 Feature 생성(Ex) PCA)

Feature Selection

기존 Feature에서 원하는 Feature만 선택하는 방법

Review
마무리

Data Preprocessing

- Data Cleansing(결측치, 이상치)
- Scaling/Encoding(Standard ~ Normalizer, Label ~ Target encoding)
- Feature Extraction(PCA)
- Feature Selection

2023 D&A

ML Session 4차시

THANK YOU

2023. 09. 26