2주차 세션 복습

훈련 세트와 테스트 세트

데이터 전처리

1. 훈련 세트와 테스트 세트

- 지도 학습 / 비지도 학습

- 훈련 세트/ 테스트 세트

✓ 지도 학습 / 비지도 학습

지도 학습(Supervised Learning)

- 샘플 데이터(input)에 대한 정답(target)을 가지고 학습
- 머신러닝 알고리즘에 입력과 기대되는 정답을 제공

분류 (classification)

회귀 (regression)

비지도 학습(Unsupervised Learning)

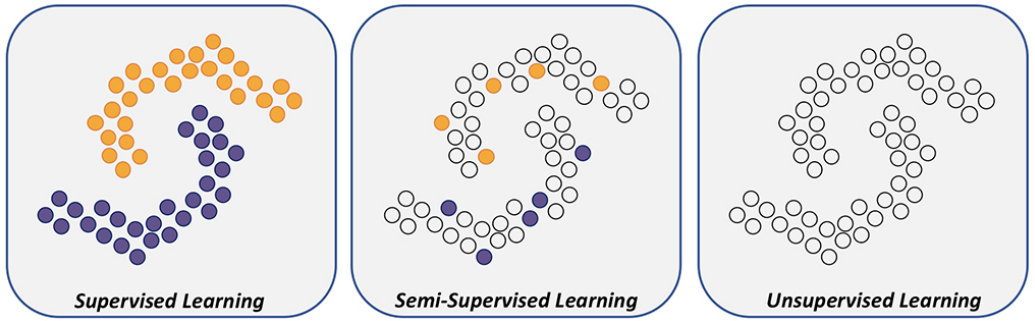
- 정답(target)없이 샘플 데이터(input)만 가지고 학습
- 특정 출력을 낼 수 없음
- 데이터 구성을 파악하는데 도움

군집 (clustering)

패턴 인식

준지도 학습(semi-supervised Learning)

- 지도 학습 + 비지도 학습
- Supservised learning은 데이터의 패턴을 외우는 학습법에 불과. 일반화가 잘되기 위해 더 많은 labeled data가 요구됨
- labeled data를 확보하기 어려운 경우 다수 존재
- 적은 labeled data가 있으면서 추가로 활용할 수 있는 대용량의 unlabeled data가 있을 경우 유용
- 데이터 자체의 본질적인 특성 가이드로 일반화 성능 향상



✓ 지도학습 - 분류 / 회귀

분류 (classification)

- 입력 데이터에 대한 가능성 있는 여러 클래스 중 하나를 예측
  - KNN
  - 결정 트리 (Decision Tree)
  - 로지스틱 회귀 (Logistic Tree)
  - SVM (SVC)

    : 커널을 이용해 독립변수들의 차원을 변환 하여 유연하지 못한 직선 및 초평면을 유연하게 하는 데 유용
  - 나이브 베이즈 (Naïve Bayes)

회귀 (regression)

- 입력 데이터에 대한 연속적인 숫자를 예측
  - 선형 회귀(Linear regression)
    - 릿지(Ridge) : L2 규제
    - 라쏘(Lasso) : L1 규제
    - 엘라스틱넷(Elastic-Net) : L1 + L2 규제
  - KNN, SVM(SVR) 등도 회귀에 이용 가능

✓ 비지도학습 – 비지도 변환 / 군집

비지도 변환

- Scaler
  - 표준화(Standardization) : 정규분포로 만듦
    - StandardScaler
  - 정규화(Normalization) : 특정 범위(주로 [0,1])로
    - MinMaxScaler
- 차원 축소
  - PCA (주성분 분석) : 공분산 이용, 주성분끼리 독립
  - NMF (비음수 행렬 분해) : 음수를 포함하지 않는 행렬을 음수를 포함하지 않는 행렬들로 분해

군집 (Clustering)

- 데이터셋을 클러스터라는 그룹으로 나누는 작업
- K-평균 군집

  : 중심 샘플 하나로 클러스터 정의하여 둥근 형태
- 병합 군집

  : 목표 클러스터 개수가 나올 때까지 클러스터를 합쳐 나감.

  계층적 군집 형성하여 덴드로그램으로 시각화 가능
- DBSCAN (밀도 기반 클러스터링)

  : 클러스터 개수를 지정해주지 않아도 됨

✓ 훈련 세트 / 테스트 세트

훈련 세트 (train)

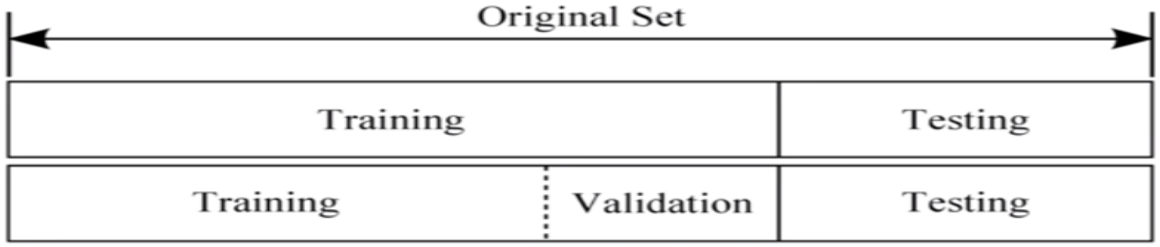
- 모델을 학습시킬 때 사용됨
- 주어진 데이터에서 70~80%를 훈련 세트로 사용
- 테스트 세트보다 많아야 함

검증 세트 (validation)

- **학습**이 이미 완료된 모델을 검증하기 위한 dataset이다.
- 훈련 세트 내에서 일부를 검증에 사용. 학습에 사용되지 않음

테스트 세트 (test)

- **학습**과 **검증**이 완료된 모델의 성능을 평가하기 위한 dataset이다.
- 테스트 데이터, 테스트 세트, 홀드아웃 세트 라고 명칭
- 주어진 데이터에서 20~30 %를 테스트 세트로 사용



✓ train_test_split

sklearn.model_selection.**train_test_split**(**arrays*, *test_size=None*, *train_size=None*, *random_state=None*, *shuffle=True*, *stratify=None*) [source]

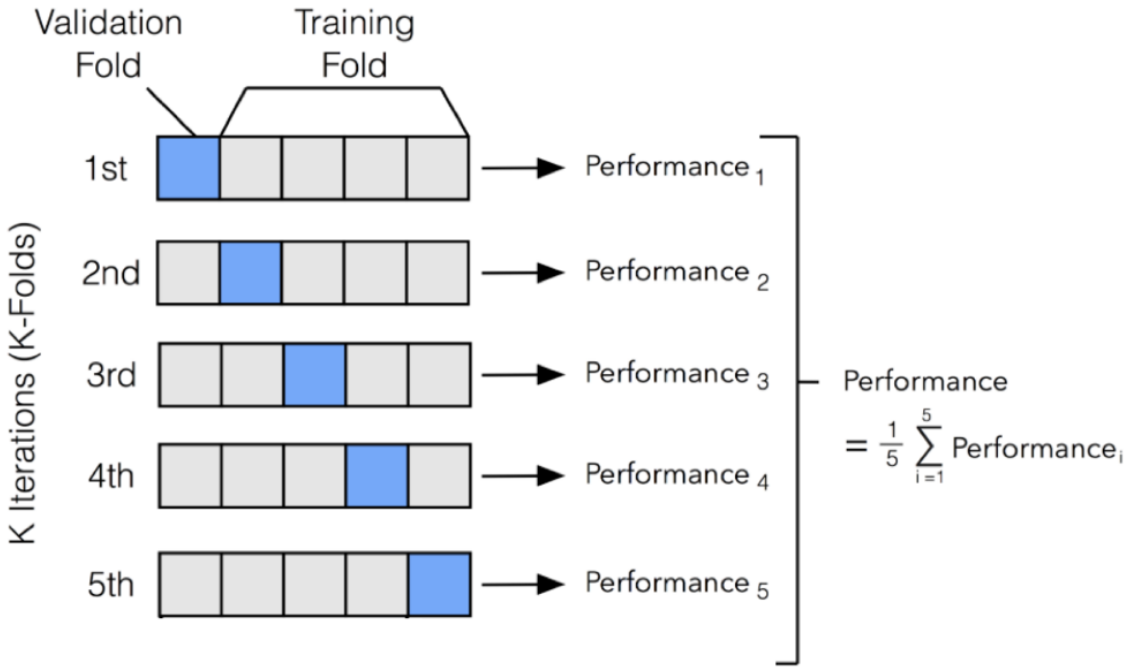
- 앞에 * 한 개가 있으면 인자를 여러 개 받을 수 있음 → 같은 크기의 array 몇 개를 인자로 넣든 train_size, test_size로 split해 줌
- Shuffle=False일 경우 데이터셋을 무작위로 섞지 않고 순차적으로 분할 함 (ex. 시계열 데이터의 경우 순서 유지)
- Stratify로 설정한 array의 클래스 비율을 유지한채 train, test로 split해 줌
- (Default) train_size = 0.75 / test_size = 0.25

✓ Cross Validation - KFold

```
sklearn.model_selection.KFold(n_splits=5, *, shuffle=False, random_state=None)
```

- 교차 검증(cross validation) :

  train set / test set을 나누는 방법 중 하나로 split을 여러 번 하여 성능 평가 지표를 평균 내어 한 번 나누는 train_test_split 보다 일반화된 성능을 얻을 수 있다.



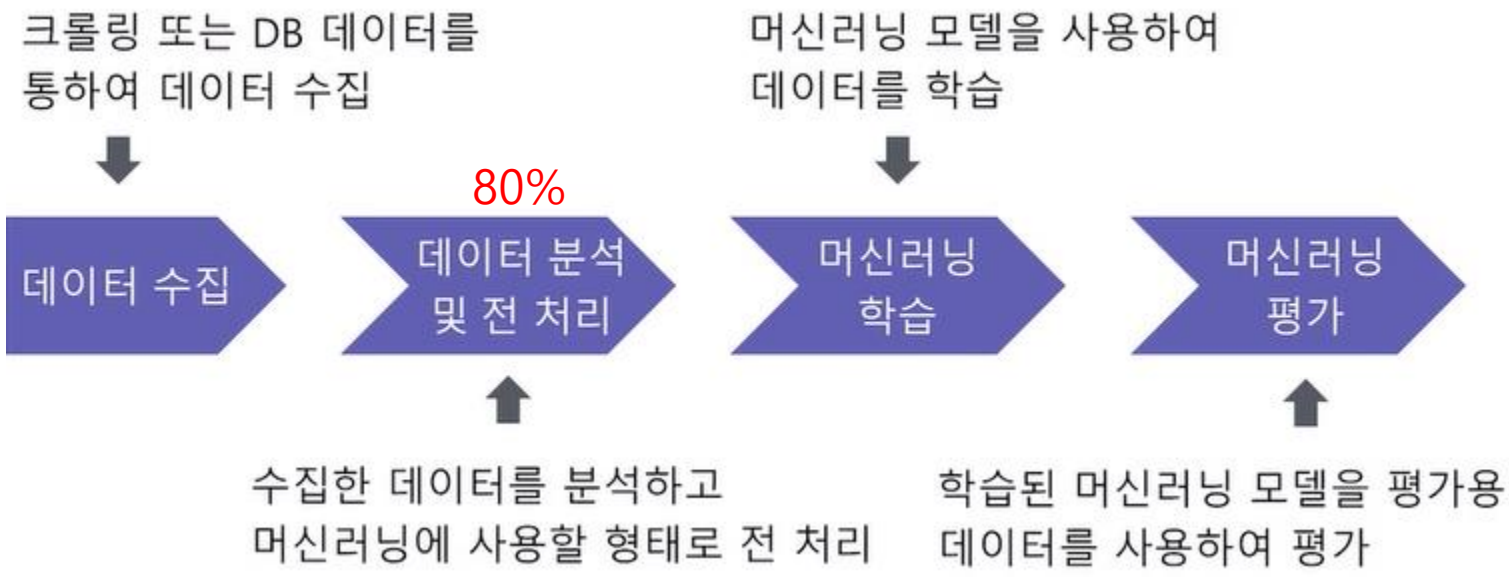
2. 데이터 전처리

- 데이터 확인
- 결측치
- 이상치
- 중복값
- 범주형 encoding
- Scaling

✓ 데이터 전처리

- 데이터 전처리란?

  왜곡된 분석결과를 방지하기 위해 분석에 적합하게 데이터를 가공하여 데이터의 품질을 올리는 일련의 과정



- 데이터 분석가에게 수집부터 평가까지 걸리는 시간의 80%가 데이터 전처리에 소요됨

✓ df.info() / df.describe()

- info() 함수는 데이터에 대한 전반적인 정보를 나타냄.
  df 구성하는 행과 열의 크기, 컬럼을 구성하는 값의 자료형 등 출력

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1704 entries, 0 to 1703
Data columns (total 8 columns):
country      1704 non-null object
continent    1704 non-null object
year         1704 non-null int64
lifeExp      1704 non-null float64
pop          1704 non-null int64
gdpPercap    1704 non-null float64
iso_alpha    1704 non-null object
iso_num      1704 non-null int64
dtypes: float64(2), int64(3), object(3)
memory usage: 106.6+ KB
```

- describe() 함수는 데이터의 컬럼별 요약 통계량을 나타냄
  숫자형 컬럼의 평균, 분산, 최대, 최소, 사분위값들을 출력

	year	lifeExp	pop	gdpPercap	iso_num
count	1704.00000	1704.000000	1.704000e+03	1704.000000	1704.000000
mean	1979.50000	59.474439	2.960121e+07	7215.327081	425.880282
std	17.26533	12.917107	1.061579e+08	9857.454543	248.305709
min	1952.00000	23.599000	6.001100e+04	241.165877	4.000000
25%	1965.75000	48.198000	2.793664e+06	1202.060309	208.000000
50%	1979.50000	60.712500	7.023596e+06	3531.846989	410.000000
75%	1993.25000	70.845500	1.958522e+07	9325.462346	638.000000
max	2007.00000	82.603000	1.318683e+09	113523.132900	894.000000

✓ df.head() / df.tail()

- head() 함수는 데이터프레임의 맨위에서부터 인자로 받는 값의 개수만큼 출력 (default=5)

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4

- tail() 함수는 데이터프레임의 맨아래에서부터 인자로 받는 값의 개수만큼 출력 (default=5)

	total_bill	tip	sex	smoker	day	time	size
239	29.03	5.92	Male	No	Sat	Dinner	3
240	27.18	2.00	Female	Yes	Sat	Dinner	2
241	22.67	2.00	Male	Yes	Sat	Dinner	2
242	17.82	1.75	Male	No	Sat	Dinner	2
243	18.78	3.00	Female	No	Thur	Dinner	2

✓ 결측치 확인

- 결측치란?

  대부분의 머신러닝 알고리즘은 NaN, 즉 누락된 데이터가 있을 때, 제대로 역할을 하지 못함.

  그래서 먼저 NaN에 대해 처리 필요.

  - Missing feature, NA(Not Available), NaN, Null, None

- 결측치 여부

```
df["col"].isnull()
```

- 결측치 개수

```
df["col"].isnull().value_counts()
```

✓ 결측치 처리

- 삭제 / 제거 : dropna()

➢ axis (default=0) : 0일시 행 삭제, 1일시 열 삭제

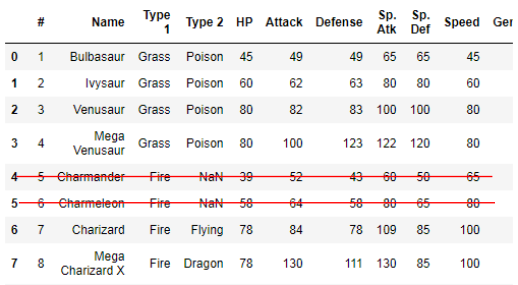
➢ subset : 특정 columns을 지정하여 해당 column 결측치 제거

➢ how (default : 'any') : 'any'일시 1개라도 NaN일 때 drop

'all'일시 모두 NaN일 때 drop

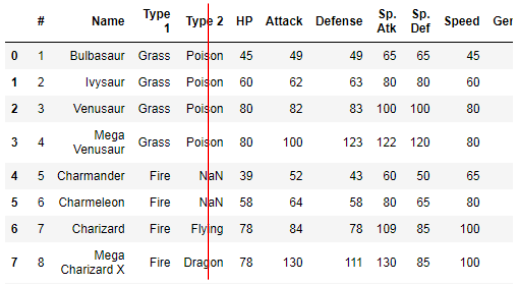
➢ thresh : 결측치가 아닌 것들의 개수가 임계값 미만 시 삭제

1. 행 삭제



	#	Name	Type 1	Type 2	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Ger
0	1	Bulbasaur	Grass	Poison	45	49	49	65	65	45	
1	2	Ivysaur	Grass	Poison	60	62	63	80	80	60	
2	3	Venusaur	Grass	Poison	80	82	83	100	100	80	
3	4	Mega Venusaur	Grass	Poison	80	100	123	122	120	80	
4	5	Charmander	Fire	NaN	39	52	43	60	50	65	
5	6	Charmeleon	Fire	NaN	58	64	58	80	65	80	
6	7	Charizard	Fire	Flying	78	84	78	109	85	100	
7	8	Mega Charizard X	Fire	Dragon	78	130	111	130	85	100	

2. 열 삭제



	#	Name	Type 1	Type 2	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Ger
0	1	Bulbasaur	Grass	Poison	45	49	49	65	65	45	
1	2	Ivysaur	Grass	Poison	60	62	63	80	80	60	
2	3	Venusaur	Grass	Poison	80	82	83	100	100	80	
3	4	Mega Venusaur	Grass	Poison	80	100	123	122	120	80	
4	5	Charmander	Fire	NaN	39	52	43	60	50	65	
5	6	Charmeleon	Fire	NaN	58	64	58	80	65	80	
6	7	Charizard	Fire	Flying	78	84	78	109	85	100	
7	8	Mega Charizard X	Fire	Dragon	78	130	111	130	85	100	

- 대치 / 보간 : fillna()

➢ value : 원하는 값으로 보간

➢ method : 'pad','ffill' : 이전 행의 값

'bfill','backfill' : 다음 행의 값

- 주로 사용하는 대체 값

평균값 : df.mean()

중간값 : df.median()

최빈값 : df.mode(), value_counts()

0

분위수 : df.quantile(), df.describe()

주변값 : ffill, bfill

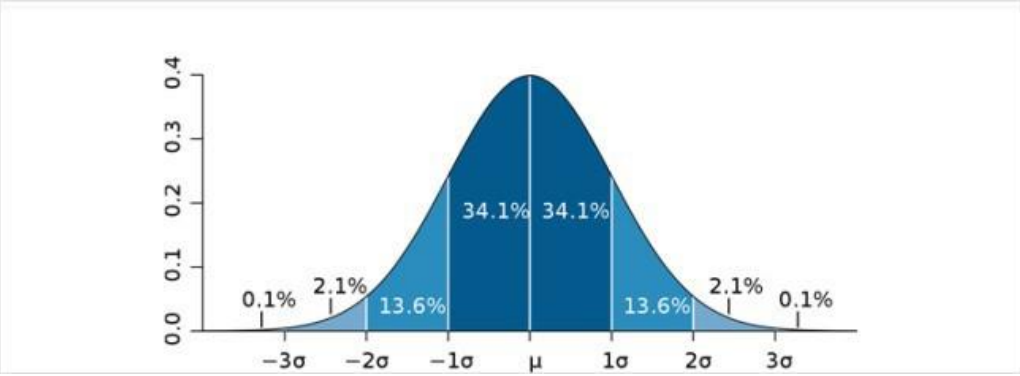
예측값 : 모델링을 통해서 특정 NaN값을 예측

✓ 이상치 처리

- 이상치 : 보통 관측된 데이터의 **범위에서 많이 벗어난** 아주 작은 값이나 큰 값을 말함
- 작거나 큰 값의 기준?

1. 데이터가 정규분포를 따를 때

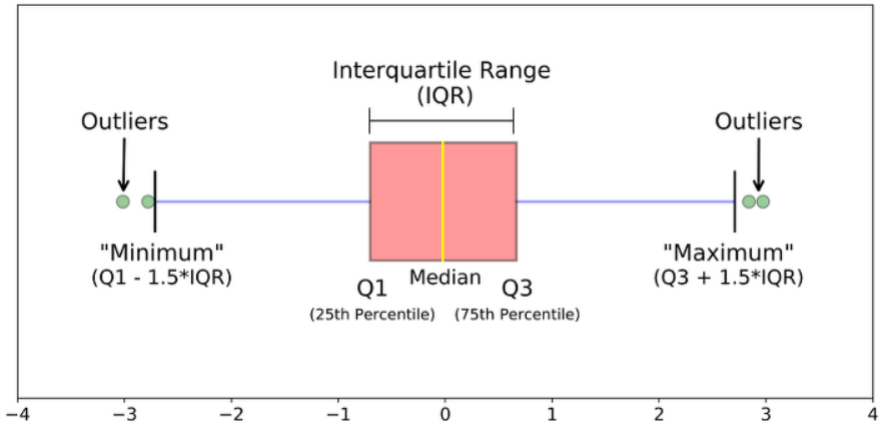
   쳐 졌을 때



표준편차(σ)를 기준으로

주로 ±(3*σ) 바깥에 있는 값들을 이상치로 판단

2. 데이터가 정규분포를 이루지 않고 치우



주로 Q1-(1.5*IQR) 미만, Q3+(1.5*IQR) 초과일 때 이상치로 판단

= 정규분포일 때 2.698 * σ와 동일

✓ 중복값 처리

- 중복값 확인 : duplicated()

➢ subset :
column들을 지정할 경우 그 columns들만 고려
미지정시 모든 columns 고려

➢ keep (default : 'first') :
'first'일 때 맨 처음 중복 행 빼고 True 반환
'last'일 때 맨 마지막 중복 행 빼고 True 반환
False일 때 모든 중복 행 True 반환

- 중복값 제거 : drop_duplicates()

➢ subset :
column들을 지정할 경우 그 columns들만 고려
미지정시 모든 columns 고려

➢ keep (default : 'first') :
'first'일 때 맨 처음 중복 행만 남기고 나머지 제거
'last'일 때 맨 마지막 중복 행만 남기고 나머지 제거
False일 때 모든 중복 행 제거

✓ Label Encoding

- 레이블 인코딩?
  범주형 데이터를 수치형으로 변환해주는 기법

1. sklearn.preprocessing.**LabelEncoder**
2. pandas.factorize
3. df.astype('category').cat.codes
4. sklearn.preprocessing.OneHotEncoder
5. pandas.get_dummies

3. df.astype('category').cat.codes

➢ cat : Series.str와 같이 pandas에는 문자열, 범주형 등의 데이터에 특화된 특수 메소드 존재

➢ codes : 수치형으로 encoding

```
1  cat_s
```

```
0    a
1    b
2    c
3    d
4    a
5    b
6    c
7    d
dtype: category
Categories (4, object): ['a', 'b', 'c', 'd']
```

```
1  cat_s.cat.categories
```

```
Index(['a', 'b', 'c', 'd'], dtype='object')
```

```
1  cat_s.cat.codes
```

```
0    0
1    1
2    2
3    3
4    0
5    1
6    2
7    3
dtype: int8
```

코드 한 줄로 끝낼 수 있어서 간단함

✓ Label Encoding

5. pandas.get_dummies

- OneHotEncoding을 매우 쉽게 해줌 :
  2차원 데이터, to_array 등의 변환이 필요한 OneHotEncoding과 달리 데이터 프레임 그 자체로 입출력

```
df=pd.get_dummies(df,columns=['column'],prefix='column') #df에 자동으로 추가됨
```

- columns : 변환 할 columns입력
- prefix (default : 원래 column name) : 변환 후 새로 생성될 column의 name 앞부분 원하는 String으로 고정

```
pd.get_dummies(fruit, columns=['name'])   #dataframe에서 특정 열만 인코딩
```

	color	name_apple	name_banana	name_cherry	name_durian
0	red	1	0	0	0
1	yellow	0	1	0	0
2	red	0	0	1	0
3	green	0	0	0	1
4	NaN	0	0	0	0

✓ Feature Scaling

- Feature Scaling ?

  독립변수의 범위 또는 데이터 특성을 정규화 및 표준화하는 데 사용되는 방법

- Why?

  1. 변수 값의 범위 및 단위가 달라서 발생 가능한 문제 예방
  2. 머신러닝 모델이 특정 데이터의 편향성을 갖는 것을 방지

- 종류

  1. 표준화 : sklearn.preprocessing.StandardScaler()

     서로 다른 범위의 변수들을 평균이 0이고 분산이 1인 가우시안 정규 분포로변환하는 작업

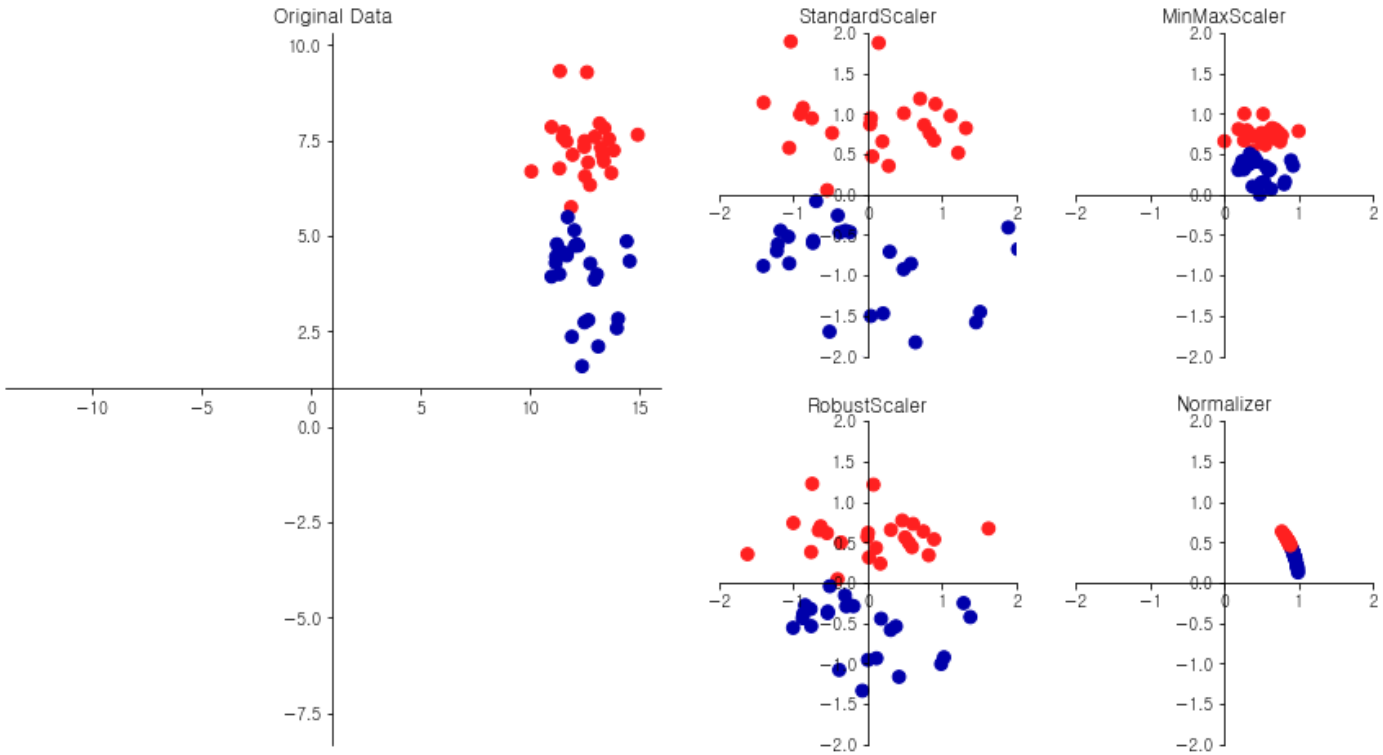
     표준화를 진행한 후 이상치를 찾아서 제거하는 게 바람직

  2. 정규화 : sklearn.preprocessing.MinMaxScaler()

     서로 다른 범위의 변수들의 크기를 통일하기 위해 이를 변환하는 작업. 일반적으로 [0, 1] 범위의 분포로 조정

✓ Feature Scaling

- 스케일링 예시



- RobustScaler : 평균 분산 대신 중간 값, 사분위 값 사용 / 이상치에 영향을 받지 않음
- Normalizer : 특성 벡터의 유클리디안 길이가 1이 되도록 데이터 포인트를 조정 (지름이 1인 원(구)에 투영) / 데이터의 방향(각도)만 중요할

감사합니다

By. 토게토게