Data Collection and Preprocessing

2025. 8

Yongjin Jeong, KwangWoon University

Data Collection and Acquisition

Data Collection and Acquisition

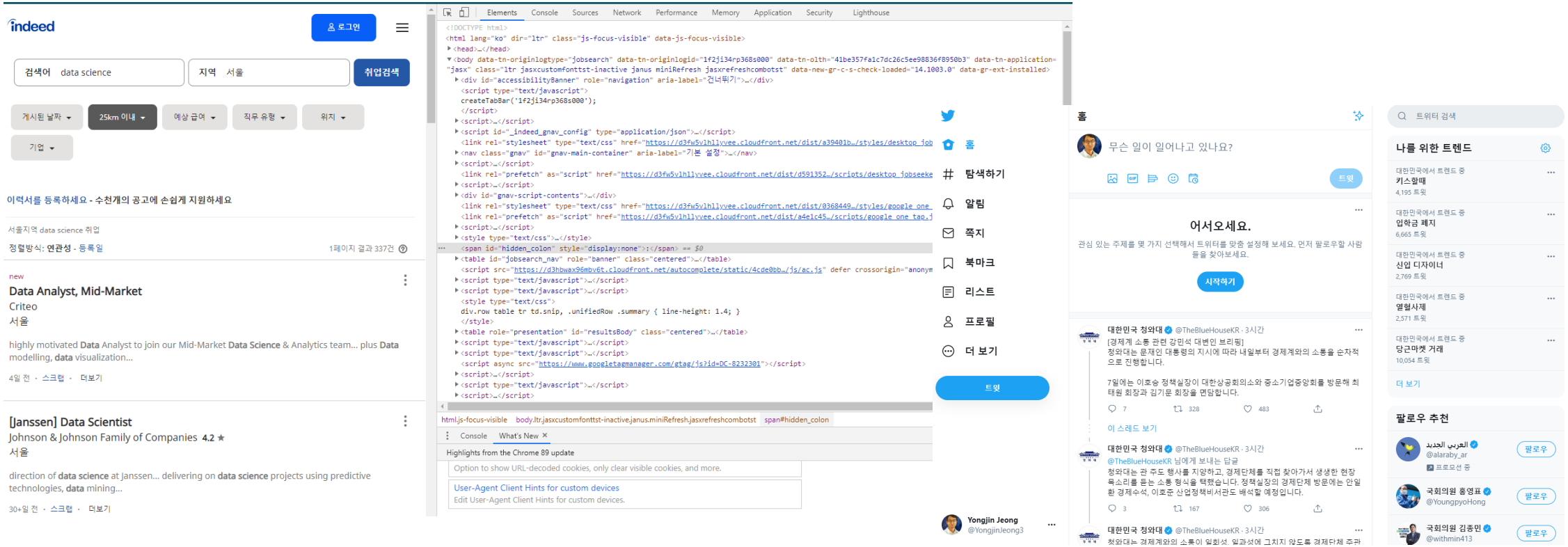
- **Collecting well-structured data**
  - ✓ String
  - ✓ CSV format: pd.read_csv(), csv.reader(), csv.writer()
  - ✓ JSON format: json.loads(), json.dumps(), pd.io.json_normalize()

```
"Month", "1958", "1959", "1960"
"JAN", 340, 360, 417
"FEB", 318, 342, 391
"MAR", 362, 406, 419
"APR", 348, 396, 461
"MAY", 363, 420, 472
"JUN", 435, 472, 535
"JUL", 491, 548, 622
"AUG", 505, 559, 606
"SEP", 404, 463, 508
"OCT", 359, 407, 461
"NOV", 310, 362, 390
"DEC", 337, 405, 432
```

```
>>> data = [{'state': 'Florida',
...          'shortname': 'FL',
...          'info': {'governor': 'Rick Scott'},
...          'counties': [{'name': 'Dade', 'population': 12345},
...                       {'name': 'Broward', 'population': 40000},
...                       {'name': 'Palm Beach', 'population': 60000}]},
...         {'state': 'Ohio',
...          'shortname': 'OH',
...          'info': {'governor': 'John Kasich'},
...          'counties': [{'name': 'Summit', 'population': 1234},
...                       {'name': 'Cuyahoga', 'population': 1337}]}]
>>> result = pd.json_normalize(data, 'counties', ['state', 'shortname',
...                                               ['info', 'governor']])
>>> result
         name  population    state shortname info.governor
0        Dade       12345   Florida    FL    Rick Scott
1     Broward       40000   Florida    FL    Rick Scott
2  Palm Beach       60000   Florida    FL    Rick Scott
3      Summit        1234   Ohio       OH    John Kasich
4    Cuyahoga        1337   Ohio       OH    John Kasich
```

Data Collection and Acquisition

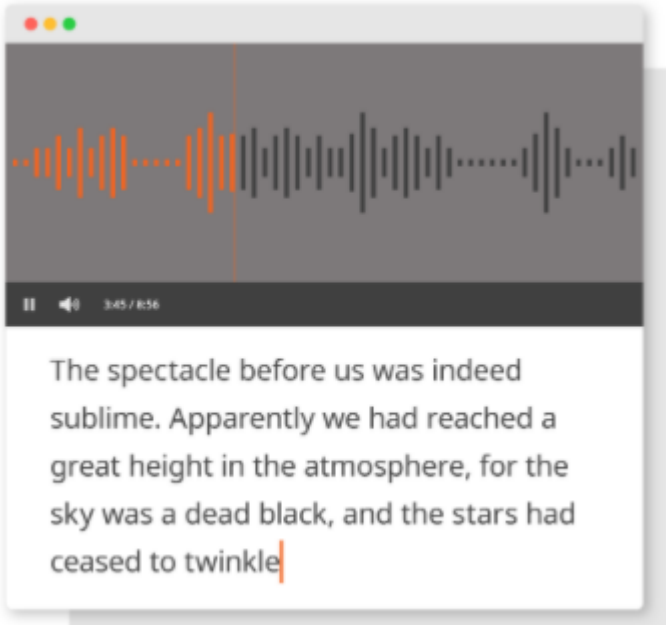
- **Acquisition from Web and SNS (scraping)**
  - ✓ Web scraping (html format): BeautifulSoup(), lxml.html()
  - ✓ SNS scraping (text processing):



Data Collection and Acquisition

- **Collecting complex and unstructured data**
  - ✓ Text: CountVectorizer(), WordVec(), Embedding layer
  - ✓ audio (and speech), video
  - ✓ Natural language

	type	text
0	ham	Go until jurong point, crazy.. Available only ...
1	ham	Ok lar... Joking wif u oni...
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...
3	ham	U dun say so early hor... U c already then say...
4	ham	Nah I don't think he goes to usf, he lives aro...
5	spam	FreeMsg Hey there darling it's been 3 week's n...
6	ham	Even my brother is not like to speak with me. ...
7	ham	As per your request 'Melle Melle (Oru Minnamin...
8	spam	WINNER!! As a valued network customer you have...
9	spam	Had your mobile 11 months or more? U R entitle...



Data Preprocessing

Data Preprocessing (전처리)

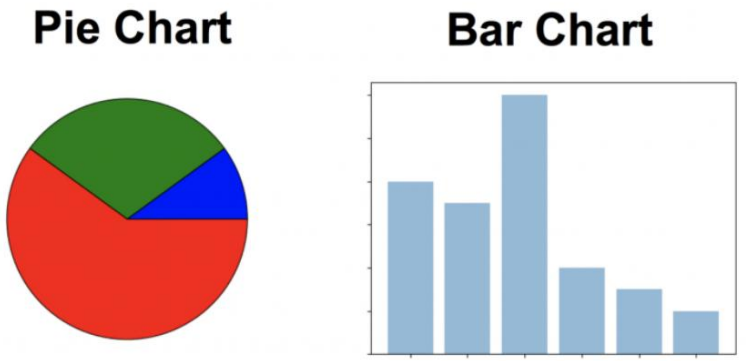
- **데이터 전처리의 중요성**
  - GIGA(Gabage In, Garbage Out): 데이터의 품질이 분석 및 모델 성능을 좌우한다.
  - 머신러닝 파이프라인
    - [데이터 수집] → **[데이터 전처리]** → [탐색적 데이터 분석(EDA)] → [모델링] → [평가 및 배포]
    - 데이터 전처리' 단계가 모델 성능에 미치는 영향이 매우 크다.
  - 전처리된 데이터의 특징: 잘 전처리된 데이터는 모델이 패턴을 학습하기 용이한 형태를 가진다.
    - 모델 성능 극대화
    - 데이터의 신뢰성 및 일관성 확보
    - 과적합(Overfitting) 방지 및 모델 일반화

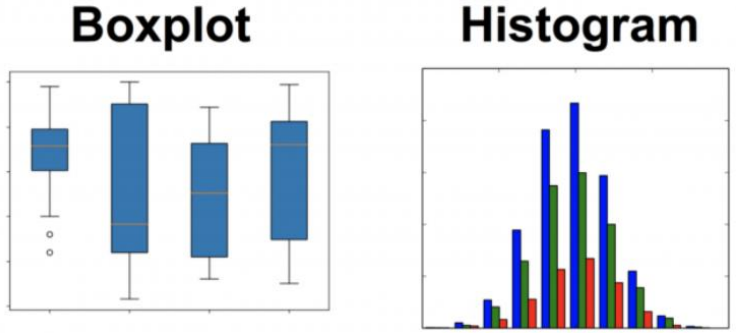
Data types (데이터 유형)

- **Quantitative (정량적):** expressed as a number (**Numerical**)
  - Discrete Data: can not subdivide, and has limited number of possible values
    - Number of students in a class, number of workers in a company, number of correct answers
  - Continuous Data: can be meaningfully divided
    - Interval Data: measurable and ordered, but have no meaningful zero
      - Can add/subtract, but can not multiply/divide/ratio
      - Because of no true zero, some descriptive and inferential statistics can not be applied
      - temperature
    - Ratios data: ordered and have a true zero
      - height, weight, length, speed
- **Qualitative (정성적):** can not be expressed as a number (**Categorical**)
  - Nominal Data: no particular order (referred as 'labels')
    - Gender (Women, Men), Hair color (Blonde, Brown, Red, etc.), Marital status (Married, Single, Widowed)
  - Ordinal Data: has particular order, but can not do arithmetic operations
    - Ranking (First, second, third, etc.), Economic status (low, medium, high), Clothes size (XL, L, M, S)

Data types and Visualization

- **Nominal data**
  - Mostly use **One-hot encoding**
  - Use a pie chart or a Bar chart to visualize
- **Ordinal data**
  - Use Label encoding (or Ordinal encoding) or One-hot encoding
  - Use Pie-chart, Bar chart, percentile, median, mode, and IQR to summarize data
- **Continuous data**
  - Histogram: can check the central tendency, variability, kurtosis, etc. but no outliers
  - Box-plot shows outliers





Data Preprocessing Steps

- **데이터 전처리 주요 단계**
  - **1. 데이터 정제 (Data Cleaning)**: 결측치, 이상치, 오류 등을 처리하여 데이터의 신뢰성을 높이는 단계.
  - **2. 특성 공학 (Feature Engineering)**: 기존 변수를 변형하거나 조합하여 모델 성능을 높일 새로운 변수를 만드는 단계
  - **3. 데이터 변환 (Data Transformation)**: 데이터를 분석하기 적합한 형태로 변환하는 단계 (스케일링, 인코딩 등)
  - **4. 데이터 축소 (Data Reduction)**: 데이터의 양을 줄여 분석의 효율성을 높이는 단계

GIGA(Gabage In, Garbage Out): 데이터의 품질이 분석 및 모델 성능을 좌우한다.

Data Cleaning

- **Missing Value Handling (결측치 저리)**
  - 제거: 행 또는 열 전체를 삭제하는 방법. 데이터가 충분히 클 때 사용 가능
  - 대체 (Imputation)
    - 수치형 데이터: 0, 평균(mean), 중앙값(median) 등으로 대체
    - 범주형 데이터: 최빈값(mode) 또는 'Other'와 같은 특정 값으로 대체
  - `Dataframe.dropna()`, `dataframe.fillna()` 등.

Data Cleaning

- **Outlier Handling (이상치 처리)**
  - 이상치 탐지
    - 시각화: Box Plot을 통해 이상치를 시각적으로 확인하는 것이 가장 효과적.
    - 통계적 방법:
      - IQR Rule: Q1−1.5×IQR 보다 작거나 Q3+1.5×IQR 보다 큰 값을 이상치로 판단.
      - Z-score: 데이터가 정규분포를 따른다고 가정할 때, Z-score가 특정 임계값(예: 2 또는 3)을 초과하는 데이터를 이상치로 판단.
  - 이상치 처리
    - 제거: 이상치를 포함하는 행을 삭제
    - 대체/조정 (Capping): 이상치를 상한(upper limit)이나 하한(lower limit) 값으로 대체

Feature Engineering (특성공학)

- **Feature Engineering**
  - 도메인 지식을 활용하여 기존 데이터로부터 모델 성능을 향상시키는 새로운 특성(feature)을 만드는 과정
  - 주요 기법:
    - 파생 변수 생성:
      - 가입일 -> 가입 연도, 가입 월, 가입 요일 등 변수 세분화
      - 출생 연도 -> 나이 계산
      - 키, 몸무게 -> BMI 지수 생성
    - Binning (구간화)::
      - 연속형 변수를 범주형 변수로 변환.
      - 예: 나이 (19, 23, 35,...) -> 연령대 ('10대', '20대','30대',...)
    - 상호작용 특성:
      - 두 개 이상의 변수를 곱하거나 나누어 새로운 상호작용을 나타내는 변수 생성

Data Transformation (데이터변환)

- 데이터 변환 1: **스케일링(Scaling)**
  - 필요성: 변수 간 값의 범위(scale)가 다를 때 이를 비슷하게 맞춰 주기 위함
  - 종류:
    - **정규화 (Normalization / Min-Max Scaling)**: $x' = \frac{x - x_{\min}}{x_{\max} - x_{\min}}$.
    - **표준화 (Standardization)**: $z = \frac{x - \mu}{\sigma}$.
    - **Robust Scaling**: $z = \frac{x - \text{median}}{IQR}$ (이상치에 덜 민감함).

- 데이터 변환 2: **Categorical Encoding**
  - Label Encoding, Ordinal Encoding, One-Hot Encoding, Binary Encoding, Frequency Encoding, Target Encoding 등 다양한 기법

Data Transformation (데이터변환)

- 데이터 변환 3: **Text Preprocessing (텍스트 전처리)**
  - 자연어(텍스트) 데이터를 벡터화하기 전, 노이즈를 제거하고 정제하는 기본 단계
  - 1. 토큰화 (Tokenization): 문장을 의미 있는 단위(단어 또는 형태소)로 나누는 과정
  - 2. 정제 (Cleaning): 불필요한 구두점, 숫자, 특수문자 등을 제거
  - 3. 불용어 (Stop words) 제거: 분석에 큰 의미가 없는 단어(a, the, is, 조사, 접사 등)를 제거.
  - 4. 정규화 (Normalization): 단어의 형태를 통일
    - 어간 추출 (Stemming): 어형이 변형된 단어에서 어간을 추출. (예: studies, studying → studi)
    - 표제어 추출 (Lemmatization): 단어의 원형(기본 사전형)을 추출. (예: am, are, is → be)

Data Transformation (데이터변환)

- 데이터 변환 4: **Text Featurization (텍스트 벡터화)**
  - 정제된 텍스트를 기계가 이해할 수 있는 숫자 벡터로 변환하는 과정
  - Bag-of-Words (BoW): 문서에 포함된 단어의 등장 횟수를 기반으로 벡터를 생성. (CounterVectorizer())
  - TF-IDF (Term Frequency-Inverse Document Frequency): 단어의 중요도를 계산하여 벡터를 생성. 모든 문서에 공통적으로 나타나는 단어의 가중치는 낮추고, 특정 문서에만 자주 나타나는 단어의 가중치는 높인다 (TfidfVectorizer())

Data Reduction (데이터축소)

- 데이터 축소 1: **특성 선택 (Feature Selection)**
  - 모델링에 사용할 가장 관련성 높은 특성만 선택하는 과정
  - 목적: 모델의 복잡도 감소, 과적합(Overfitting) 방지, 훈련 시간 단축
  - 주요 기법:
    - 상관관계 분석: 상관관계가 높은 변수 중 하나를 제거
    - 모델 기반 선택: 의사결정나무, Lasso 회귀 등 모델 자체의 특성 중요도를 활용
- 데이터 축소 2: **차원 축소 (Dimensionality Reduction)**
  - 많은 수의 변수를 정보 손실을 최소화하며 더 적은 수의 새로운 변수로 요약하는 과정.
  - 목적: '차원의 저주' 문제 해결, 데이터 시각화 용이
  - 주요 기법:
    - 주성분 분석 (PCA, Principal Component Analysis): 데이터의 분산을 가장 잘 설명하는 새로운 축(주성분)을 찾아, 원본 데이터를 해당 축에 투영시키는 기법

Data Preprocessing (보조자료)

Data Preprocessing

- **Transform raw data into an understandable format.**
- **Data in real world is:**
  - Incomplete: missing data, lacking attribute values, lacking certain attributes of interest, or containing only aggregate (총액) data
  - Noisy: containing errors (or measurement errors) and outliers
  - Inconsistent: containing discrepancies in codes or names
  - Distorted: sampling distortion
- **"No quality data, no quality mining (and data science) result!"**
  - Garbage In, Garbage Out.
  - Quality decisions must be based on quality data
  - Data warehouse needs consistent integration of quality data

Data Preprocessing

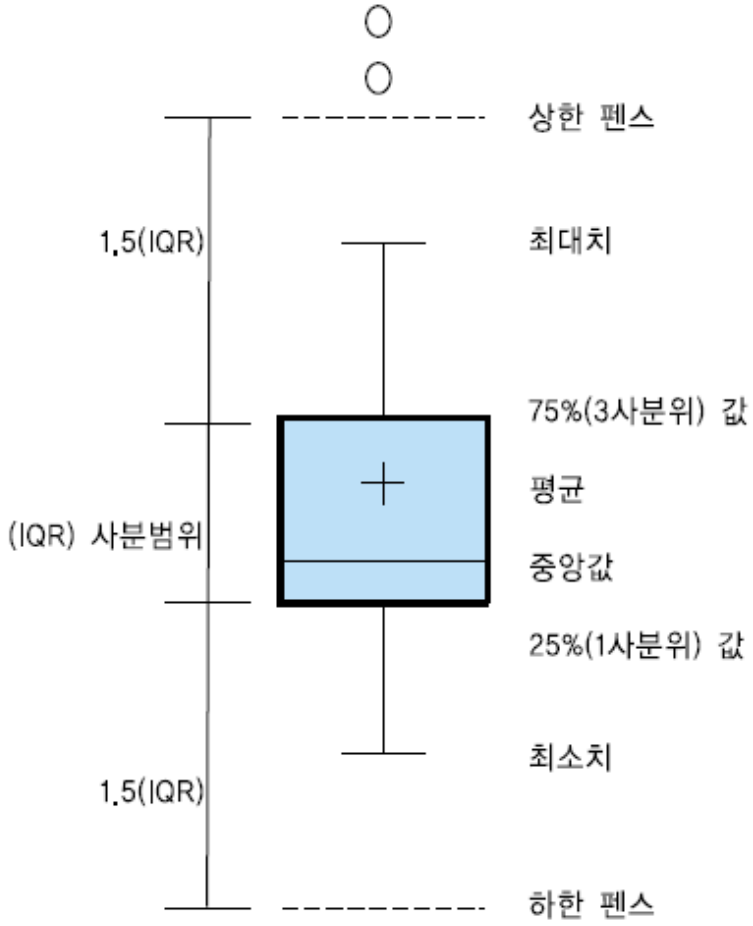
- **Feature Engineering**
  - Redefine / integrate / create / reduce features
  - (ex) built_year –> how_old, cumulative -> by_terms, total -> per_section, etc.
- **Data Cleaning**
  - ✓ Distinguishing errors (during data collection, unrecoverable) from artifacts (during data processing, recoverable)
  - ✓ Data compatibility
  - ✓ Dropping or Imputation of Missing values (결손값의 대체)
  - ✓ Estimating unobserved (zero) counts
  - ✓ Detection and processing of Outliers and Invalid values
- **Data Transformation**
  - ✓ Normalization and Scaling, Categorical Encoding, Text Featurization
- **Data Reduction**
  - ✓ Obtains reduced representation in volume but produces the same or similar analytical results

Data Preprocessing

- **Missing value handling**
  - Drop the rows or the entire column (if data size is large enough)
  - Numerical imputation (replace missing values with 0, median, mean value, max/min, etc.)
  - Categorical imputation (replacing missing values with the maximum occurred value in a column, or imputing a category like "Other")
  - dataframe.dropna(), dataframe.fillna(data.mean())
- **Invalid value handling**
  - Dropping or replacing
- **Handling Outliers**
  - Visualizing is the best way to detect outliers.
  - Statistical methodologies can also be used. (x * standard deviation, z-score, etc.)
  - Dropping or Capping (to upper and lower limit)

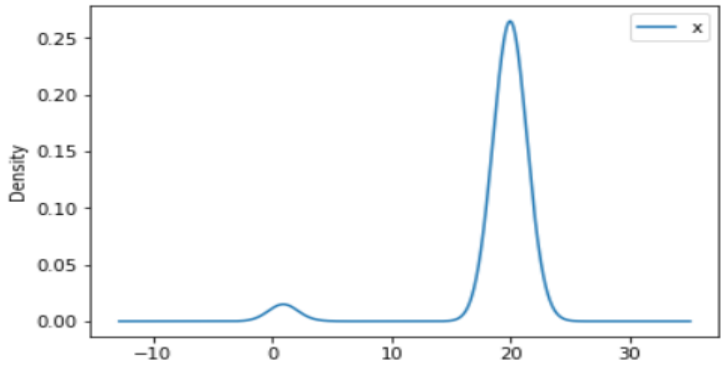
Data Preprocessing - Scaling

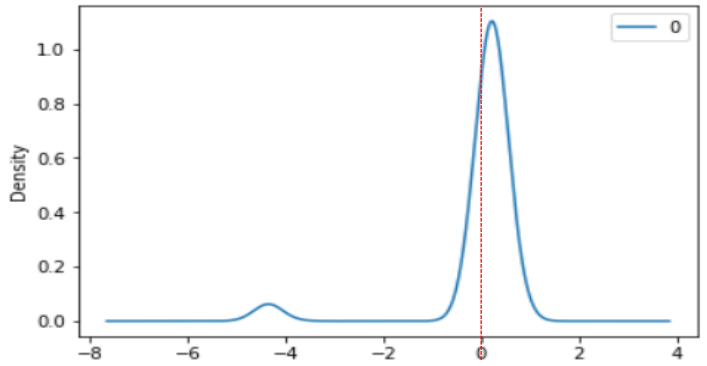
- **Why scaling?**
  - What if two columns have different ranges for the values? Hard to compare.
- **Normalization (min-max scaling):**
  - $x = (x - x_{min}) / (x_{max} - x_{min})$
- **Standardization (standard scaling):**
  - $z = (x - \mu) / \sigma$
- **Robust scaling:**
  - $z = (x - median) / IQR$
  - use median and IQR (instead of μ and σ), robust to outliers
- **Which one to use?**
  - depends on your data

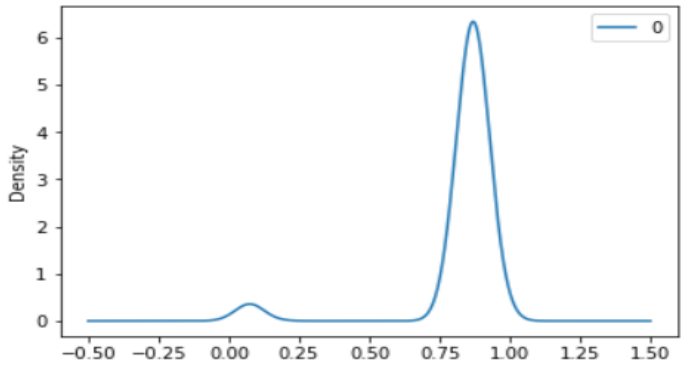


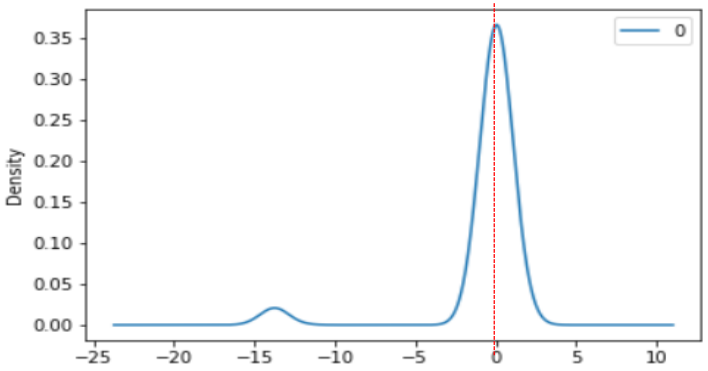
Data Preprocessing - Scaling

- **Scaling example**
  - ✓ Standard, Min-max, Robust scaling
  - ✓ `pd.DataFrame({'x':np.concatenate([np.random.normal(20, 1, 1000), np.random.normal(1, 1, 50)])`









Data Preprocessing – Categorical Encoding

- **Categorical variables**
  - ✓ **Nominal Variable** (**Categorical**): Variable comprises a finite set of discrete values with no relationship between values.
  - ✓ **Ordinal Variable:** Variable comprises a finite set of discrete values with a ranked ordering between values.
- **Nominal Encoding: —** Where Order of data does not matter
  - ✓ One Hot Encoding
  - ✓ Binary encoding
  - ✓ One Hot Encoding With Many Categories
  - ✓ Mean Encoding
- **Ordinal Encoding: —** Where Order of data matters
  - ✓ Target Guided Ordinal Encoding

Data Preprocessing – Categorical Encoding

- **Label encoding**:
  - ✓ convert the labels into numeric form (0 ~ n_classes-1)
  - ✓ Do not care whether the variables is ordinal or not
- **Ordinal encoding:**
  - ✓ Integer encoding (like Label Encoding) for features, but can specify an order
  - ✓ When there is a known relationship between the classes
  - ✓ (ex) cancer stages, earthquake magnitudes, size of shoes or clothes
- **One-hot encoding**:
  - ✓ spread the feature values to multiple flag columns and assigns 0 or 1 to them.
  - ✓ When there is no known relationship between the classes
  - ✓ (ex) colors (R, G, B), countries, species
- For more, see
  - ✓ https://towardsdatascience.com/all-about-categorical-variable-encoding-305f3361fd02

Data Preprocessing – Categorical Encoding

State (Nominal Scale)
Maharashtra
Tamil Nadu
Delhi
Karnataka
Gujarat
Uttar Pradesh

State (Label Encoding)
3
4
0
2
1
5

label encoding

Original Encoding	Ordinal Encoding
Poor	1
Good	2
Very Good	3
Excellent	4

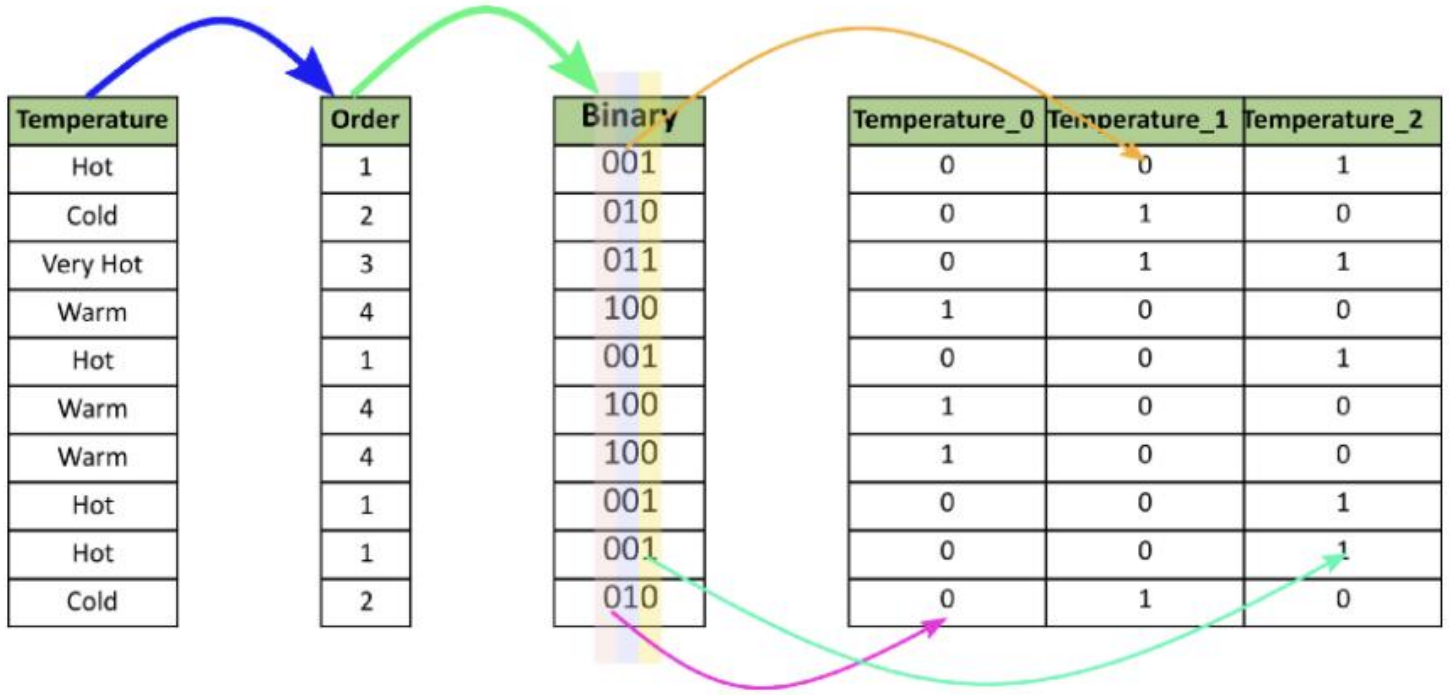
ordinal encoding

State	State_Maharashtra	State_Tamil Nadu	State_Delhi	State_Karnataka	State_Gujarat	State_Uttar Pradesh
Maharashtra	**1**	0	0	0	0	0
Tamil Nadu	0	**1**	0	0	0	0
Delhi	0	0	**1**	0	0	0
Karnataka	0	0	0	**1**	0	0
Gujarat	0	0	0	0	**1**	0
Uttar Pradesh	0	0	0	0	0	**1**

One-hot encoding

Data Preprocessing – Categorical Encoding

- **Binary Encoding**
  - converts a category into binary digits (each binary digit creates one feature column)
  - Compared to One Hot Encoding, fewer feature columns
  - (ex) for 100 categories, One Hot Encoding will have 100 features while Binary encoding will need just seven features.
  - BinaryEncoder() function in category_encoders package



Temperature
Hot
Cold
Very Hot
Warm
Hot
Warm
Warm
Hot
Hot
Cold

Order
1
2
3
4
1
4
4
1
1
2

Binary
001
010
011
100
001
100
100
001
001
010

Temperature_0	Temperature_1	Temperature_2
0	0	1
0	1	0
0	1	1
1	0	0
0	0	1
1	0	0
1	0	0
0	0	1
0	0	1
0	1	0

Data Preprocessing – Categorical Encoding

- **Frequency Encoding**
  - utilize the frequency of the categories as labels
  - Useful when the frequency is related somewhat with the target variable

- Encoding of categorical levels of feature to values between 0 and 1 based on their relative frequency

A	0.44 (4 out of 9)
B	0.33 (3 out of 9)
C	0.22 (2 out of 9)

Feature	Encoded Feature
A	0.44
A	0.44
A	0.44
A	0.44
B	0.33
B	0.33
B	0.33
C	0.22
C	0.22

Data Preprocessing – Categorical Encoding

- **Target Encoding**
  - very popular encoding approach followed by Kagglers
  - similar to label encoding, except here labels are correlated directly with the target (each category is encoded as **the mean value of the target variable** on a training data)
  - To reduce Overfitting, use K-Fold Target Encoding, Smoothing, etc.

```
data = {
    'City': ['Seoul', 'Busan', 'Incheon', 'Seoul', 'Busan'],
    'Price': [500, 450, 520, 510, 430]
}
```

target

(*) 같은 값이 나오는 경우가 있는데, 이때는 그 변수는 타겟과 아무런 상관관계가 없다는 뜻이기에 특성을 제거한다.

- Seoul의 평균 Price: (500 + 510) / 2 = 505
- Busan의 평균 Price: (450 + 430) / 2 = 440
- Incheon의 평균 Price: 520

```
Encoded Data = {
    'City': [505, 440, 520, 505, 440],
    'Price': [500, 450, 520, 510, 430]
}
```

```
import category_encoders as ce
import pandas as pd

# 예시 데이터
data = pd.DataFrame({
    'City': ['Seoul', 'Busan', 'Incheon', 'Seoul', 'Busan'],
    'Price': [500, 450, 520, 510, 430]
})

# Target Encoding 적용
target_encoder = ce.TargetEncoder(cols=['City'])
data['City_Encoded'] = target_encoder.fit_transform(data['City'], data['Price'])
```