3. 데이터 준비와 전처리의 기초

목차

- 데이터 준비와 전처리의 중요성
- 데이터 수집
- 데이터 정제
- 특성 공학 (Feature Engineering)
- 요약 및 결론

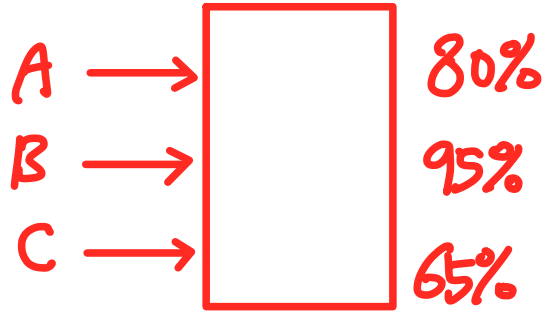
데이터의 준비와 전처리의 중요성

데이터 준비와 전처리의 중요성

- 모델의 성능 향상
  - 정확성 : 깨끗하고 잘 정제된 데이터는 모델의 정확도를 향상시킴. 노이즈를 제거하는 것이 중요함
  - 일관성 : 일관된 데이터는 모델이 일관된 패턴을 학습할 수 있도록 함.

IZM

- 데이터 품질 보장
  - 결측값 처리 : 데이터 분석과 모델 훈련 과정에서 문제 유발
  - 이상값 처리 : 모델이 잘못된 학습을 진행



- 모델의 일반화 능력 향상
  - 과적합 방지 : 모델이 특정 데이터넷에 과적합 (overfitting) 되지 않도록 함. 모델이 새로운 데이터에 대해 더 잘 일반화 (generalize) 할 수 있도록 함

데이터 준비와 전처리의 중요성

- 효율성 및 성능 최적화
  - 데이터 크기 축소 : 중복 데이터를 제거하고 불필요한 특성을 축소함으로써 모델 훈련 시간과 메모리 사용량을 감축
  - 특성 공학 : 중요한 특성을 선택하고 변환하여 모델이 더 효율적으로 학습

- 해석 가능성 향상 Explainable AI : XAI
  - 특성 선택 및 변환 : 모델의 예측을 더 잘 이해하고 설명
  - 모델의 해석 가능성을 높여 비즈니스 의사결정에 활용

- 데이터 통합 및 일관성 유지
  - 다양한 데이터 소스 통합 : 데이터 소스 간의 불일치를 해결하고 일관된 데이터셋을 구성
  - 표준화 : 다른 시스템과의 호환성을 높이고 분석 과정이 간소화

mysql. HTML.

- 모델 훈련 과정의 안정성 확보
  - 안정적인 입력 데이터 : 모델이 예측할 때 더 안정적인 성능을 발휘

데이터 수집

데이터 수집

- 데이터 수집 방법
  - 웹 스크래핑
  - **API** 사용
  - 데이터베이스에서 데이터 가져오기
  - 공개 데이터셋 활용

웹 스크래핑 (WEB SCRAPING)

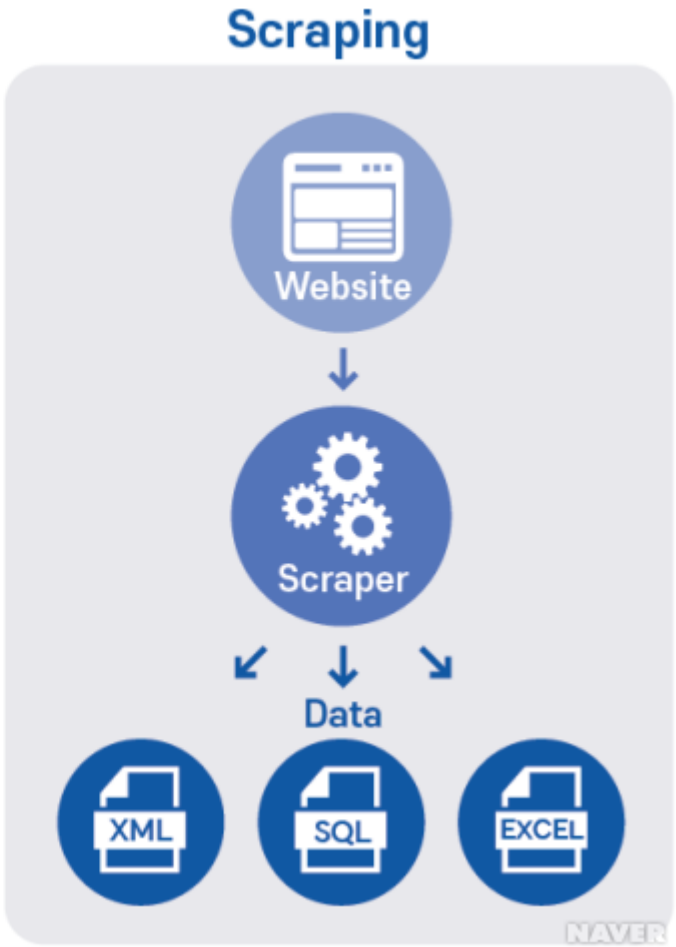
- 웹 스크래핑을 통한 데이터 수집 과정
  - 웹 사이트에서 데이터를 추출
  - 특정 웹 페이지에서 필요한 정보를 자동으로 수집

  - 목표 설정
    - 어떤 데이터를 수집할 것인지 결정
    - 필요한 데이터를 제공하는 웹 사이트 식별

  - 웹 페이지 분석
    - 웹 페이지의 구조를 분석하고 **HTML** 소스를 검토하여 필요한 데이터가 포함된 태그를 찾는다
    - 브라우저의 개발자 도구를 사용하여 **HTML** 구조를 이해

웹 스크래핑

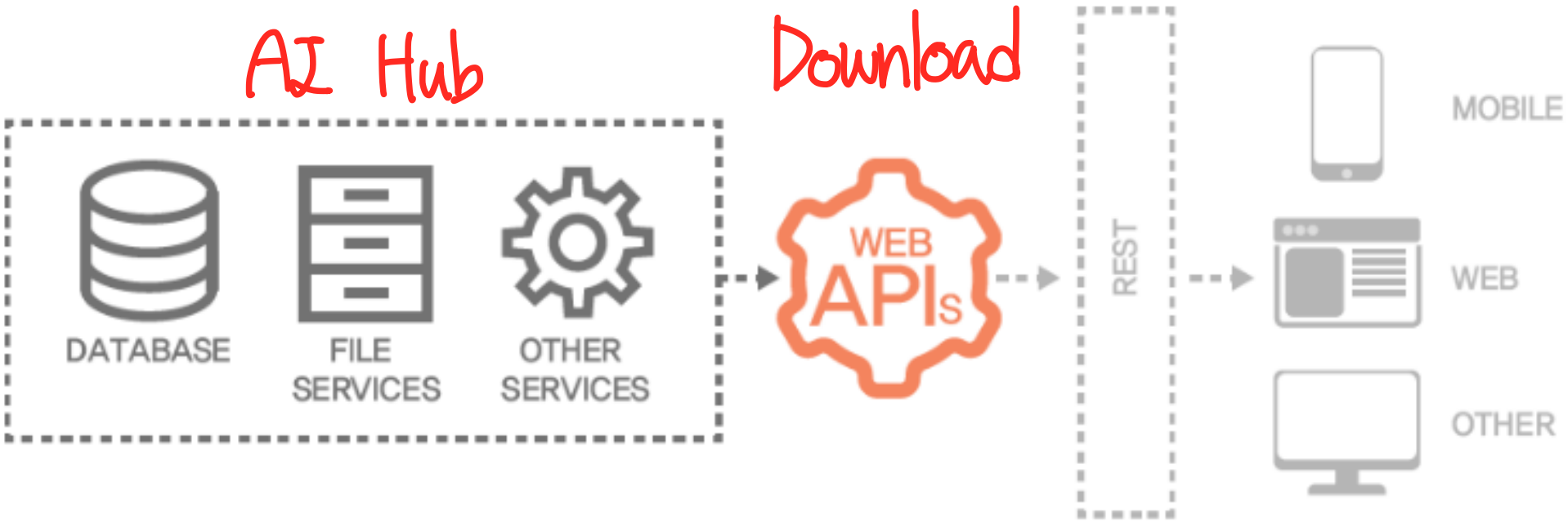
- 웹 스크래핑 **vs** 웹 크롤링
  - 웹 스크래핑
    - 특정 웹 사이트에서 필요한 데이터를 수집
    - 크롤링보다 좁은 범위의 데이터 수집
    - 온라인 쇼핑몰의 상품 정보, 뉴스 사이트의 최신 기사 등
  - 웹 크롤링 **(Web Crawling)**
    - 인터넷 상에 존재하는 모든 웹 페이지를 방문하여 데이터를 수집
    - 각 페이지의 링크를 따라가면서 자동으로 데이터를 수집
    - 대부분의 검색 엔진에서 사용



원하는 정보 추출의 정확도가 어떻게 더 높은지?

API 사용

- API 의 기본 원리



API 사용

- API (Application Programming Interface) 사용
  - 데이터 수집을 자동화하고 구조화된 방식으로 데이터를 획득

- API를 사용하여 데이터를 수집하는 과정
  - API 키 및 접근 권한 얻기
    - API 제공 사이트에 가입하고, 필요한 경우 API 키를 생성하여 접근 권한 획득
  - API 문서 읽기
    - API 매뉴얼을 숙지하여 API의 사용 방법, 엔드포인트, 파라미터, 응답 형식을 이해
    - API 매뉴얼은 일반적으로 API 제공자의 웹사이트에 게재
  - HTTP 요청 보내기
    - API는 일반적으로 HTTP 요청을 통해 데이터를 제공
    - 'request' 라이브러리를 사용하여 GET 요청을 보내고 응답 수신
  - 응답 데이터 처리
    - 일반적으로 json 형식으로 제공
    - Json 데이터를 파싱하여 필요 정보 추출

데이터베이스

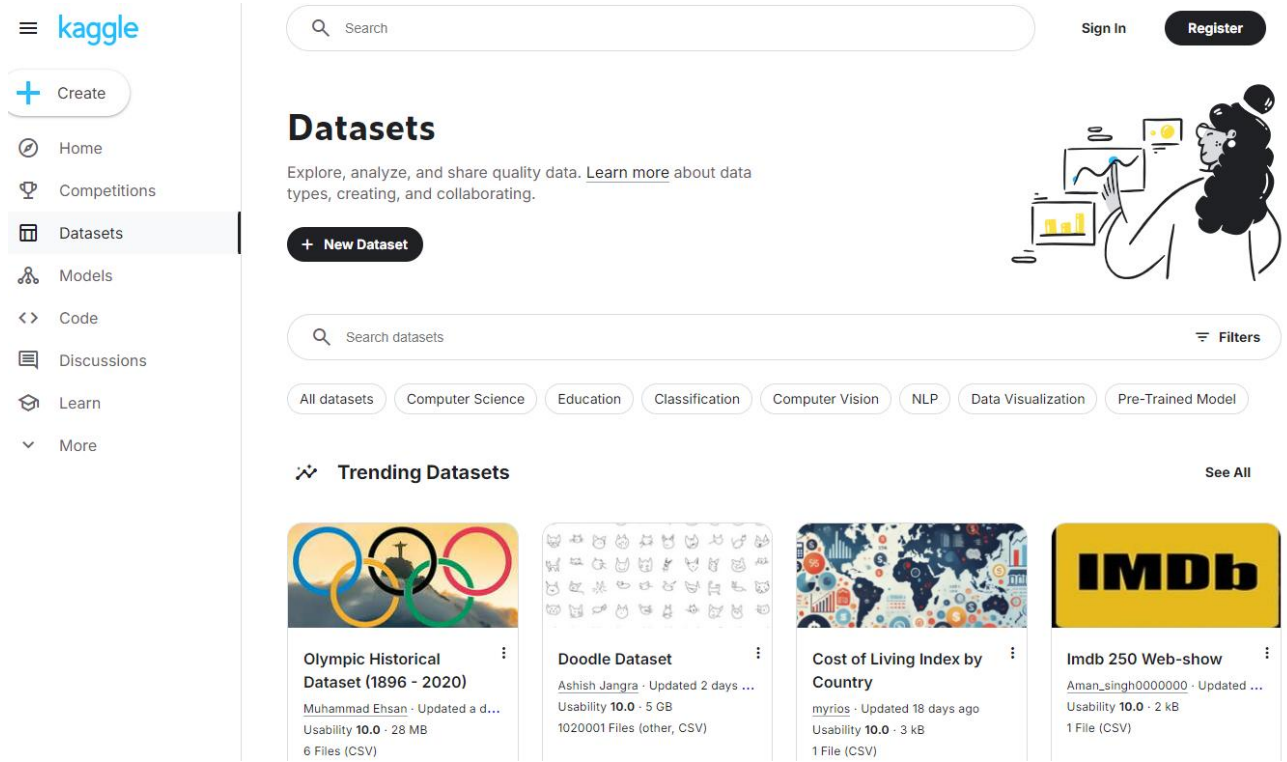
- 데이터베이스에서 데이터를 추출
  - 요구 사항 정의 및 데이터베이스 연결 설정
    - 요구사항 정의 : 목표 모델에 맞는 데이터의 종류를 정의하고 데이터베이스의 스키마를 이해
    - 데이터베이스 연결 설정 : **DBMS (MySQL, SQL Server 등)** 을 확인하고 접근 라이브러리 확인
  - **SQL 쿼리 작성 및 실행**
    - **SQL** 쿼리 작성 : 필요한 데이터를 선택하고 필터링하여 가져올 수 있는 쿼리 작성
    - **SQL** 쿼리 실행 : 쿼리를 실행하고 결과를 가져옴
  - 데이터 전처리
    - 결측값 처리, 데이터 형식 변환, 이상값 탐지 및 제거
  - 특성 공학
    - 특성 선택, 특성 생성
  - 데이터 정규화 및 스케일링
  - 데이터 분할
  - 모델 학습 및 평가

공개 데이터셋

광개 SW, GPT-3

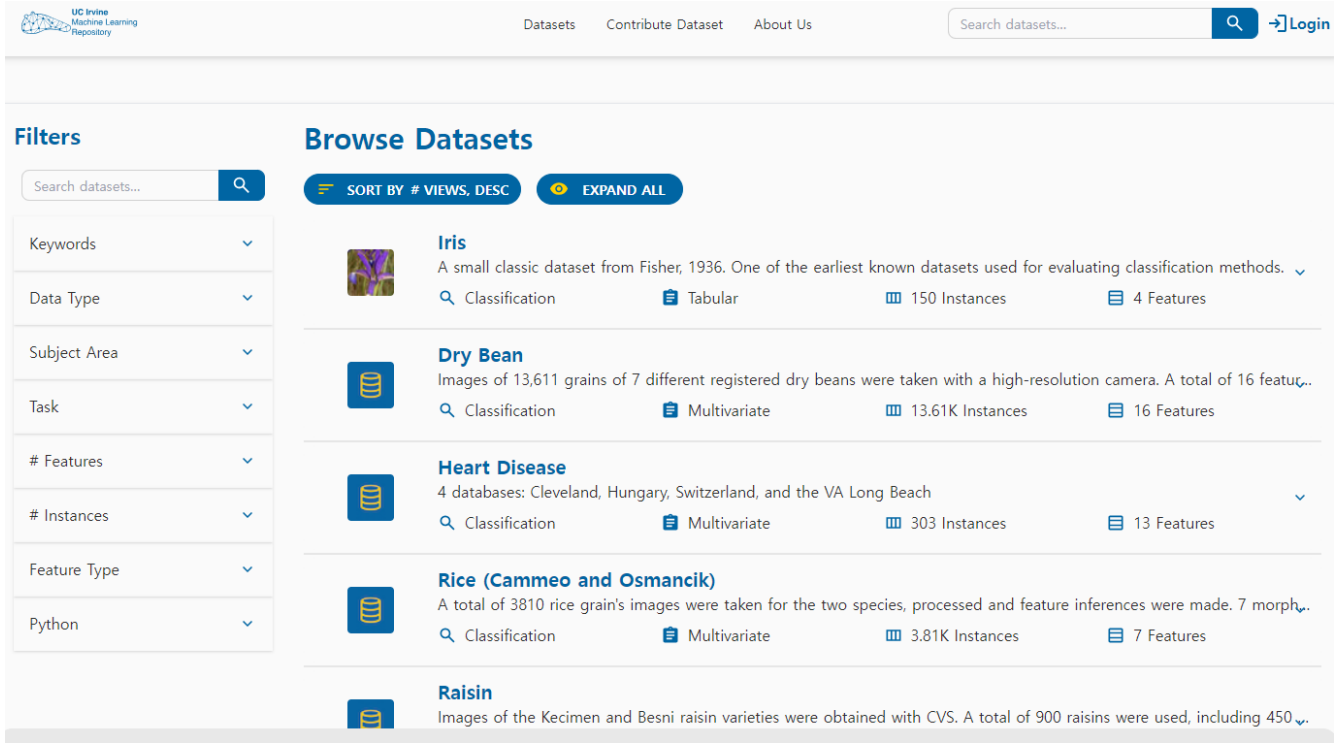
- Kaggle Datasets
  - 데이터 과학 대회 플랫폼
  - 다양한 주제의 데이터셋을 제공
  - 예시 데이터셋
    - Titanic: Machine Learning from Disaster
    - House Prices – Advanced Regression Techniques
    - IMDB: Movie Reviews Dataset

보안상의 이유가 존재한다?



공개 데이터셋

- UCI Machine Learning Repository
  - 가장 오래된 공개 데이터셋 리포지토리
  - 예시 데이터셋
    - Iris Dataset
    - Wine Quality Dataset
    - Adult Income Dataset



데이터 정제

데이터 정제의 필요성

행정구역별(읍면동)	연령별	2015	
		남자 (명)	여자 (명)
서울특별시	0~4세	197,029	187,201
	5~9세	189,058	178,865
	10~14세	207,475	193,844
	15~19세	277,732	265,351
	20~24세	340,607	340,592

데이터 정제

시점	행정구역별(읍면동)	연령별	항목	데이터
2015	서울특별시	0~4세	남자 (명)	197,029
2015	서울특별시	0~4세	여자 (명)	187,201
2015	서울특별시	5~9세	남자 (명)	189,058
2015	서울특별시	5~9세	여자 (명)	178,865
2015	서울특별시	10~14세	남자 (명)	207,475
2015	서울특별시	10~14세	여자 (명)	193,844
2015	서울특별시	15~19세	남자 (명)	277,732
2015	서울특별시	15~19세	여자 (명)	265,351

결측값 처리

- 결측값 (**Missing Value**) 사례

	A	B	C	D	E	F	G	H	I	J	K	L
1	Passenger	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
2	1	0	3	Braund, M	male	22	1	0	A/5 21171	7.25		S
3	2	1	1	Cumings,	female	38	1	0	PC 17599	71.2833	C85	C
4	3	1	3	Heikkinen	female	26	0	0	STON/O2.	7.925		S
5	4	1	1	Futrelle, M	female	35	1	0	113803	53.1	C123	S
6	5	0	3	Allen, Mr.	male	35	0	0	373450	8.05		S
7	6	0	3	Moran, Mi	male		0	0	330877	8.4583		Q
8	7	0	1	McCarthy,	male	54	0	0	17463	51.8625	E46	S
9	8	0	3	Palsson, M	male	2	3	1	349909	21.075		S
15	14	0	3	Andersson	male	39	1	5	347082	31.275		S
16	15	0	3	Vestrom, M	female	14	0	0	350406	7.8542		S
17	16	1	2	Hewlett, M	female	55	0	0	248706	16		S
18	17	0	3	Rice, Mast	male	2	4	1	382652	29.125		Q
19	18	1	2	Williams, I	male		0	0	244373	13		S
20	19	0	3	Vander Pl	female	31	1	0	345763	18		S
21	20	1	3	Masselmai	female		0	0	2649	7.225		C
22	21	0	2	Fynney, M	male	35	0	0	239865	26		S

결측값

결측값 처리

- 결측값의 영향
  - 변수 간의 관계가 부정확하게 측정됨
  - 모델의 정확성이 떨어짐

- 결측값 처리 방법
  - 삭제
    - 결측값이 무작위로 일부만 발생할 경우
    - 결측값을 포함한 자료를 삭제
  - 대체
    - 다른 값으로 대체
    - 최빈값 (가장 많이 나오는 값), 평균값 등 통계로 계산한 대푯값을 결측치로 대신
  - 예측
    - 다른 속성으로 해당 값을 유추할 수 있는 경우
    - 신발 사이즈 : 키를 통해 신발 사이즈를 예측

결측값 처리

```python
import pandas as pd

# 예시 데이터프레임 생성
data = {
    'A': [1, 2, None, 4, 5],
    'B': [None, 2, 3, None, 5],
    'C': [1, None, None, 4, 5]
}

df = pd.DataFrame(data)

# 결측값 확인
print(df.isnull())
print(df.isnull().sum())
```

```
       A      B      C
0  False   True  False
1  False  False   True
2   True  False   True
3  False   True  False
4  False  False  False
A    1
B    2
C    2
dtype: int64
```

결측값 확인

```python
# 결측값이 있는 행 제거
df_dropped_rows = df.dropna()
print(df_dropped_rows)
```

```
     A    B    C
4  5.0  5.0  5.0
```

```python
# 결측값이 있는 열 제거
df_dropped_columns = df.dropna(axis=1)
print(df_dropped_columns)
```

```
Empty DataFrame
Columns: []
Index: [0, 1, 2, 3, 4]
```

결측값 제거

```python
# 열의 평균값으로 결측값 대체
df_filled_mean = df.fillna(df.mean())
print(df_filled_mean)
```

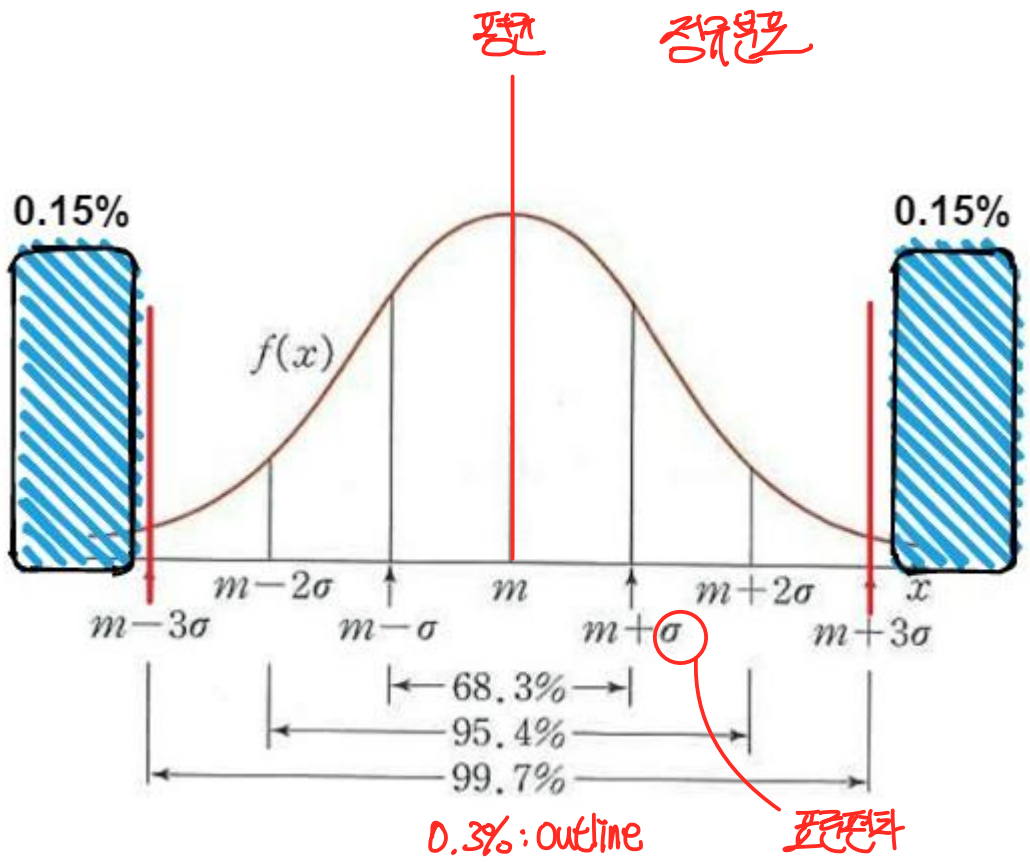
```
     A         B         C
0  1.0  3.333333  1.000000
1  2.0  2.000000  3.333333
2  3.0  3.000000  3.333333
3  4.0  3.333333  4.000000
4  5.0  5.000000  5.000000
```

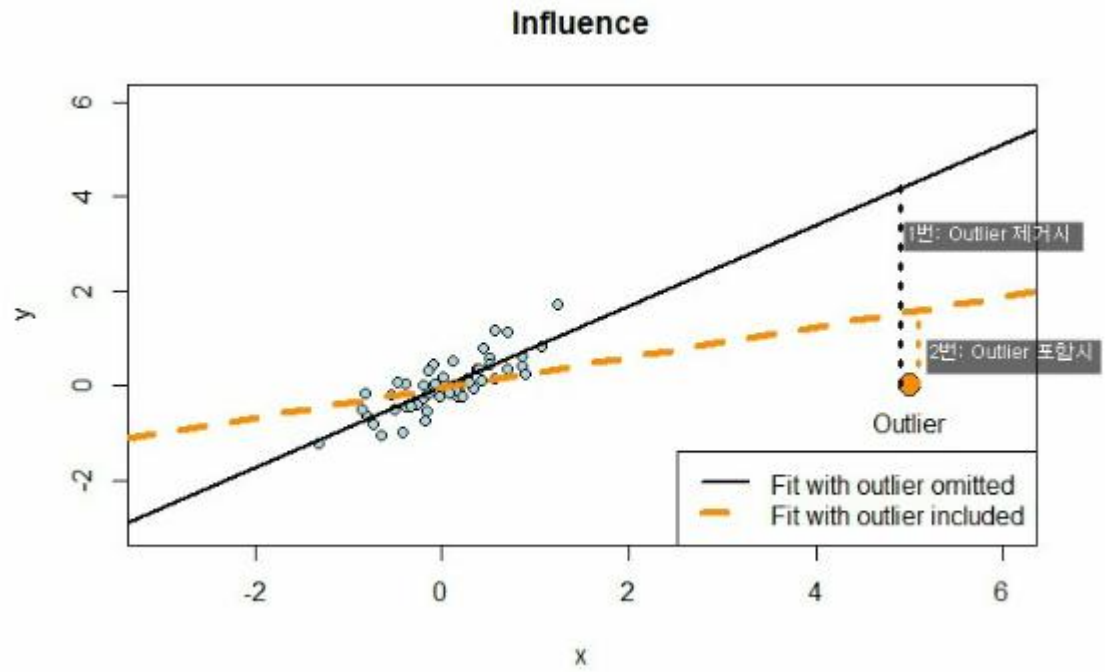
```python
# 열의 최빈값으로 결측값 대체
df_filled_mode = df.fillna(df.mode().iloc[0])
print(df_filled_mode)
```

```
     A    B    C
0  1.0  2.0  1.0
1  2.0  2.0  1.0
2  1.0  3.0  1.0
3  4.0  2.0  4.0
4  5.0  5.0  5.0
```

결측값 대체

이상값 탐지 및 처리


0.15%
0.15%
$f(x)$
$m-3\sigma$
$m-2\sigma$
$m-\sigma$
m
$m+\sigma$
$m+2\sigma$
$m+3\sigma$
x
68.3%
95.4%
99.7%
0.3% : outline


Influence
y
x
-2
0
2
4
6
1번: Outlier 제거시
2번: Outlier 포함시
Outlier
Fit with outlier omitted
Fit with outlier included

이상값 탐지 및 처리

- 이상값 처리 방법
  - 제거
    - 데이터셋이 크고 이상값이 적은 경우 데이터셋에서 이상값을 제거
  - 대체
    - 평균, 중앙값, 또는 특정 값으로 대체
  - 변환
    - 로그 변환, 제곱근 변환, **Box-Cox** 변환 등으로 값을 변환
  - 분리 및 분석
    - 이상값을 분리하여 별도로 분석

중복 데이터의 처리

	A	B	C	D	E
1	No.	상품코드	상품명	제조사	단가
2	1	NP0010	네임펜F (중간글씨용) 흑색	모나미	6000
3	2	NP0011	더블에이 A4용지	더블에이	20000
4	3	NP0012	데스크 오거나이저	하이브	15000
5	4	NP0013	모나미 볼펜	모나미	300
6	5	NP0014	보드마카 청색	동아	4300
7	6	NP0012	데스크 오거나이저	하이브	15000
8	7	NP0015	사무용 스테플러침 (33호)	피스코리아	950
9	8	NP0016	스카치 다용도 테이프	3M	900
10	9	NP0017	오피스 수정테이프	아이비스	7500
11	10	NP0018	옥스포드 노트	브랜빌	6000
12	11	NP0019	옵텍스 형광펜 혼합3색	동아	3000
13	12	NP0015	사무용 스테플러침 (33호)	피스코리아	950
14	13	NP0020	카카오프렌즈 인덱스 노트 네오	바른손	5000
15	14	NP0021	카카오프렌즈 인덱스 노트 라이언	바른손	5000
16	15	NP0022	포스트잇 노트 (654) 노랑	3M	1700

중복

중복

중복 데이터의 처리

```python
import pandas as pd

# 예시 데이터프레임 생성
data = {
    'A': [1, 2, 2, 4, 5],
    'B': ['a', 'b', 'b', 'd', 'e'],
    'C': [10, 20, 20, 40, 50]
}

df = pd.DataFrame(data)
print("Original DataFrame:")
print(df)

# 중복 데이터 확인
duplicates = df.duplicated()
print("\nDuplicated rows:")
print(duplicates)

# 중복된 행 표시
duplicated_rows = df[df.duplicated()]
print("\nDuplicated rows:")
print(duplicated_rows)
```

```python
# 중복 데이터 제거
df_no_duplicates = df.drop_duplicates()
print("\nDataFrame without duplicates:")
print(df_no_duplicates)

# 'A' 열을 기준으로 중복 데이터 제거
df_no_duplicates_a = df.drop_duplicates(subset=['A'])
print("\nDataFrame without duplicates based on column 'A':")
print(df_no_duplicates_a)

# 마지막 중복 데이터 유지
df_keep_last = df.drop_duplicates(keep='last')
print("\nDataFrame keeping the last occurrence of duplicates:")
print(df_keep_last)
```

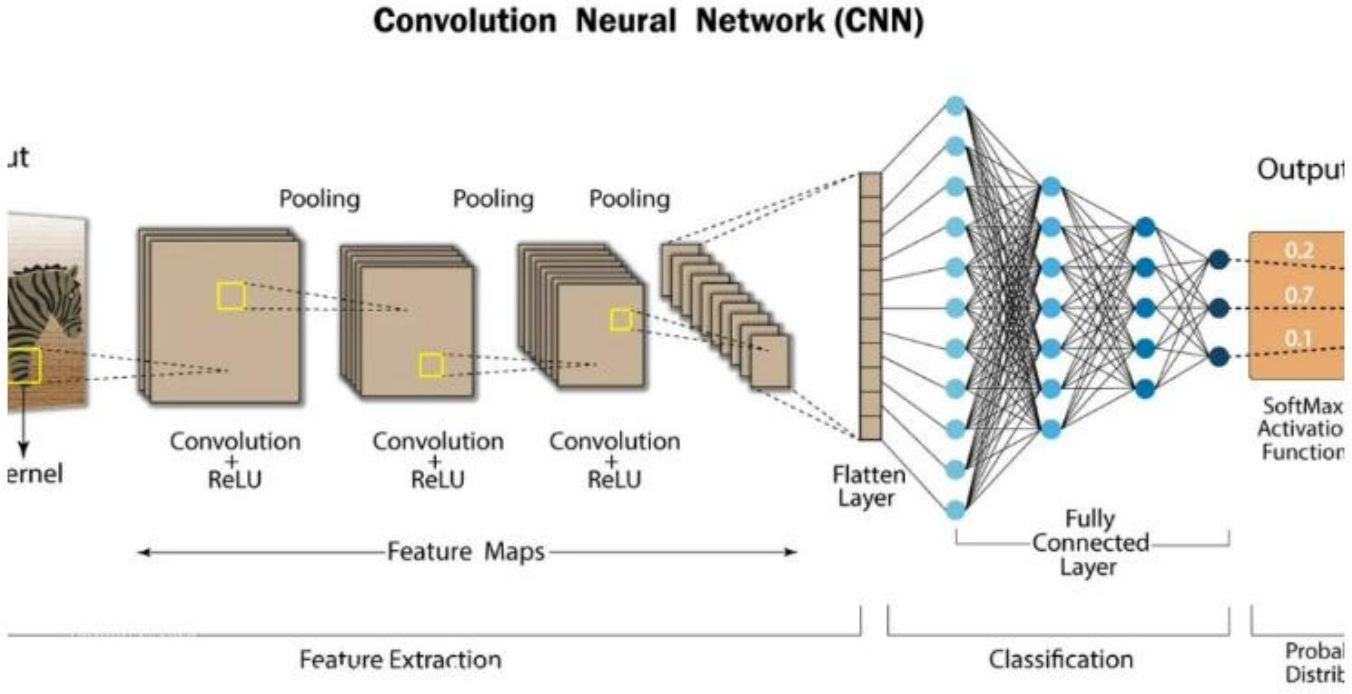
```
Original DataFrame:
   A  B   C
0  1  a  10
1  2  b  20
2  2  b  20
3  4  d  40
4  5  e  50

Duplicated rows:
0    False
1    False
2     True
3    False
4    False
dtype: bool

Duplicated rows:
   A  B   C
2  2  b  20

DataFrame without duplicates:
   A  B   C
0  1  a  10
1  2  b  20
3  4  d  40
4  5  e  50

DataFrame without duplicates based on column 'A':
   A  B   C
0  1  a  10
1  2  b  20
3  4  d  40
4  5  e  50

DataFrame keeping the last occurrence of duplicates:
   A  B   C
0  1  a  10
2  2  b  20
3  4  d  40
4  5  e  50
```

특성 공학

특성 공학

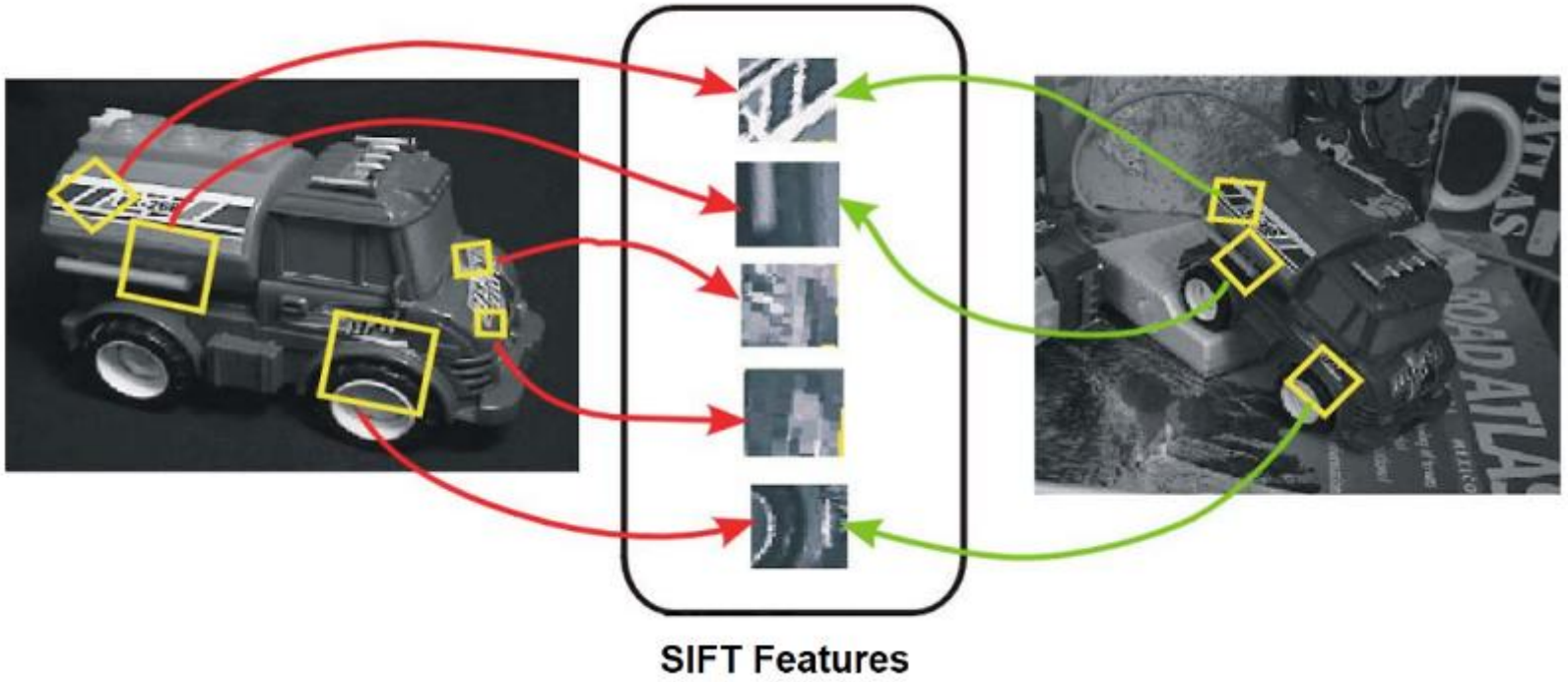
image → array
matrix

- 특성 공학 (Feature Engineering)
  - 머린 러닝 모델의 성능을 향상시키기 위해 데이터의 특성을 만들겨나 변형하는 과정
  - 데이터의 특성을 더 잘 나타냄
  - 모델에 적합한 특성을 만들어 모델의 성능을 향상



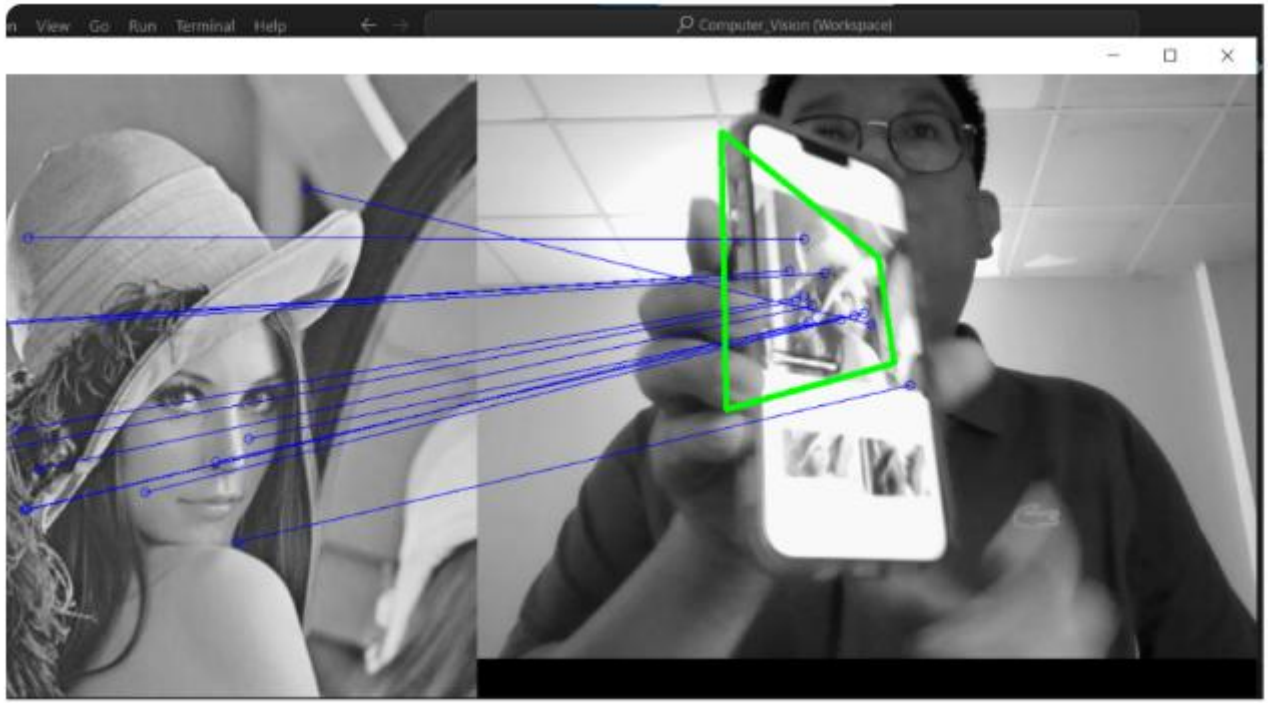
고전 특성 공학

- 이미지 인식에서의 고전적인 특성 추출 방법
  - **SIFT (Scale-Invariant Feature Transform)**



고전 특성 공학

- SURF (Speeded-Up Robust Features)



특성 변환

- 스케일링의 필요성

관측개체 번호	나이 (X_1)	월별 소득 (X_2)
1	30	3,620,000
2	13	0
3	21	600,000
4	61	500,000
5	7	0
⋮	⋮	⋮

특성 변환

- 표준화 (Standardization) vs. 정규화 (Normalization)

0~1 0~1

Normalization	Standardization
스케일링 시 최대, 최소값이 사용된다	스케일링 시 평균과 표준편차가 사용된다
피처의 크기가 다를 때 사용한다	평균이 0, 표준편차가 1인 것을 확인하고 싶을 때 (그렇게 만들고 싶을 때) 사용한다
[0,1] (또는 [-1,1]) 사이의 값으로 스케일링	특정 범위로 제한되지 않는다
분포에 대해 모를 때 유용하다	피처가 정규분포(가우시안 분포)인 경우 유용하다
MinMaxScaler, Normalizer	StandardScaler, RobustScaler

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}}$$

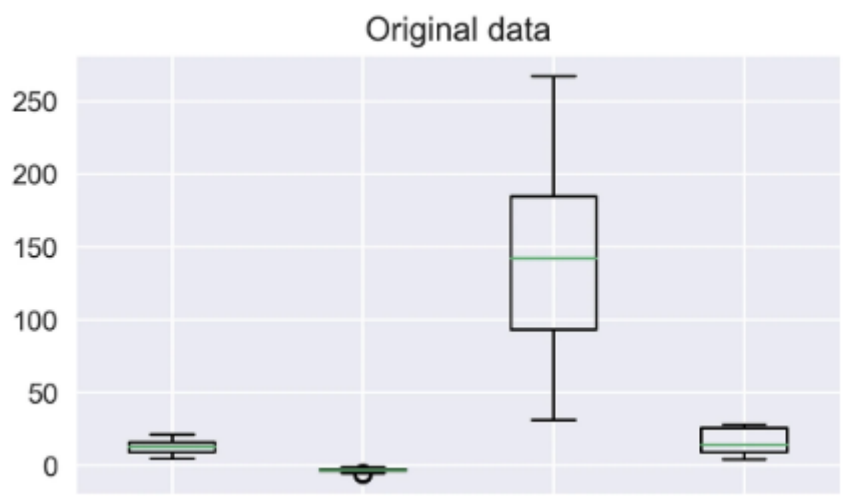
$$z = \frac{x - \mu}{\sigma}$$

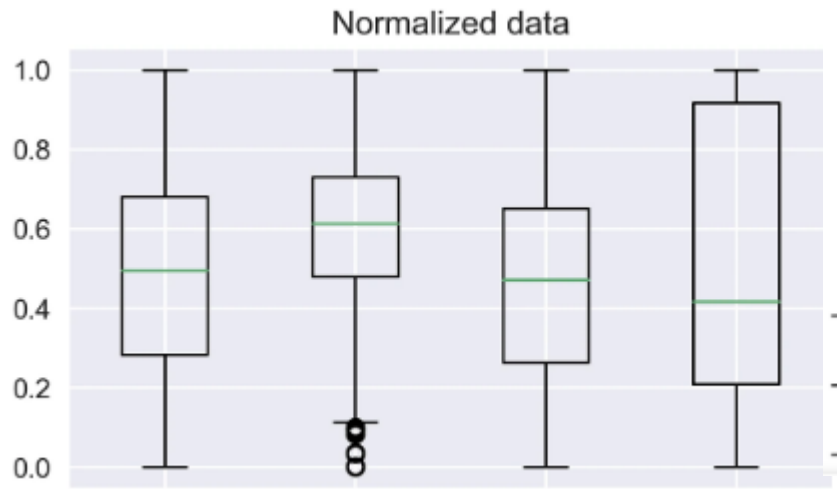
평균

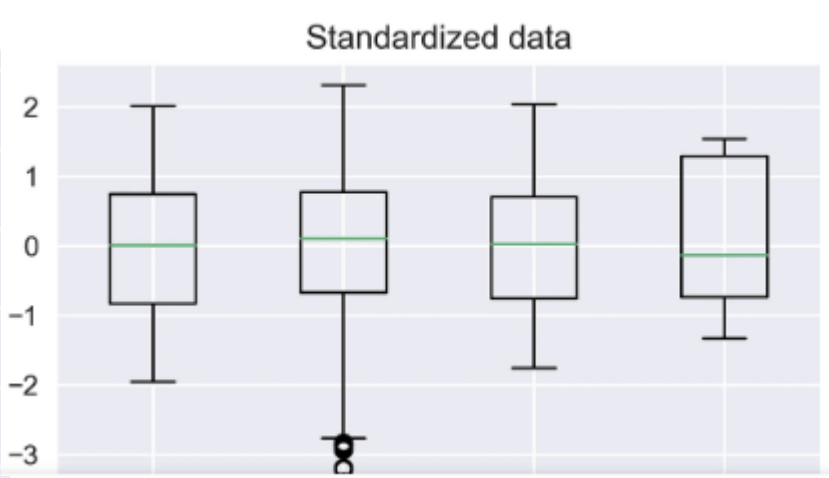
표준편차

특성 변환

- 스케일링 변환 결과







범주형 데이터 인코딩

- 범주형 데이터
  - 값이 명확한 범주 또는 그룹으로 나누어지는 데이터
  - 수치적 의미가 없고, 각 값은 고유한 범주를 표현
  - 명목형 데이터 **(Nominal Data)**
    - 범주 간에 순서나 계급이 없는 데이터
    - 색상 (빨강, 파랑, 초록), 도시 (뉴욕, 런던, 토쿄), 성별 (남성, 여성) 등
  - 순서형 데이터 **(Ordinal Data)**
    - 범주 간에 순서나 계급이 있는 데이터
    - 교육 수준 (고등학교, 학사, 석사, 박사), 고객 만족도 (매우 불만족, 불만족, 만족, 매우 만족) 등

범주형 데이터 인코딩

- 원-핫 인코딩

단어	단어 인덱스	원-핫 벡터
you	0	[1, 0, 0, 0, 0, 0, 0]
say	1	[0, 1, 0, 0, 0, 0, 0]
goodbye	2	[0, 0, 1, 0, 0, 0, 0]
and	3	[0, 0, 0, 1, 0, 0, 0]
I	4	[0, 0, 0, 0, 1, 0, 0]
say	5	[0, 0, 0, 0, 0, 1, 0]
hello	6	[0, 0, 0, 0, 0, 0, 1]

범주형 데이터 인코딩

- 레이블 인코딩

Label Encoding

Food Name	Categorical #	Calories
Apple	1	95
Chicken	2	231
Broccoli	3	50

텍스트 전처리

자연어처리 (NLP)

- 텍스트 전처리
  - 풀고자 하는 문제의 용도에 맞게 텍스트를 사전에 처리하는 작업
  - 텍스트가 제대로 전처리 되어 있지 않으면 자연어 처리 기법들이 제대로 동작하지 않음

  - 토큰화 (Tokenization)
    - 단어 토큰화 (Word Tokenization)
    - 문장 토큰화 (Sentence Tokenization)
  - 정규화 (Normalization)
    - 소문자 변환 (Lowercasing)
    - 불용어 제거 (Stopword Removal)
    - 표제어 추출 (Lemmatization) 과 어간 추출 (Stemming)
  - 정제 (Cleaning)
    - 특수 문자 제거 (Romoving Special Characters)
    - 숫자 제거 (Removing Numbers)
    - 공백 제거 및 정리 (Removing Extra Whitespace)

전처리 요약

```python
import re
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.stem import PorterStemmer

# NLTK 데이터 다운로드 (최초 한 번만 실행)
nltk.download('punkt')
nltk.download('stopwords')

def preprocess_text(input_string):
    # 1. 소문자 변환
    text = input_string.lower()

    # 2. 구두점 제거
    text = re.sub(r'[^\w\s]', '', text)

    # 3. 숫자 제거
    text = re.sub(r'\d+', '', text)

    # 4. 토큰화
    words = word_tokenize(text)

    # 5. 불용어 제거
    stop_words = set(stopwords.words('english'))
    words = [word for word in words if word not in stop_words]

    # 6. 어간 추출
    stemmer = PorterStemmer()
    words = [stemmer.stem(word) for word in words]

    # 7. 공백 및 특수문자 제거
    text = ''.join(words)

    return text

# 예시 문장
input_sentence = "This is an example sentence, demonstrating the removal of stopwords and numbers like 123."
cleaned_sentence = preprocess_text(input_sentence)
print(cleaned_sentence)
```

```
examplsentencdemonstrremovstopwordnumberlike
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

요약 및 결론

요약

- 인공지능을 위한 데이터의 준비 과정
- 데이터 준비 및 전처리의 중요성
- 데이터의 수집
  - 웹 스크래핑, **API** 사용, 데이터베이스, 공개 데이터셋
- 데이터 정제
  - 결측값 및 이상값, 중복 데이터의 탐지 및 처리
- 특성 공학
  - 머신 러닝에서의 특성 공학
  - 특성 변환 (스케일링, 인코딩 텍스트 전처리 등)