



데이터 전처리

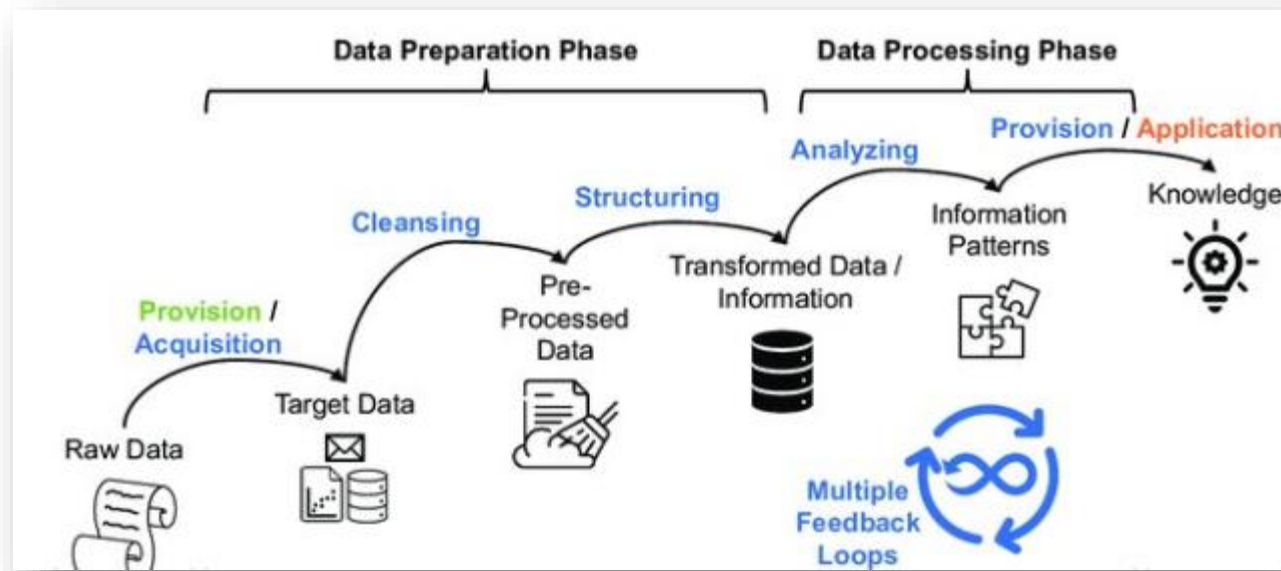
데이터 전처리

Business Understanding	Data Understanding	Data Preparation	Modeling	Evaluation	Deployment
Determine Business Objectives <i>Background</i> <i>Business Objectives</i> <i>Business Success Criteria</i>	Collect Initial Data <i>Initial Data Collection Report</i>	Select Data <i>Rationale for Inclusion/Exclusion</i>	Select Modeling Techniques <i>Modeling Technique</i> <i>Modeling Assumptions</i>	Evaluate Results <i>Assessment of Data Mining Results w.r.t. Business Success Criteria</i> <i>Approved Models</i>	Plan Deployment <i>Deployment Plan</i>
Assess Situation <i>Inventory of Resources</i> <i>Requirements, Assumptions, and Constraints</i> <i>Risks and Contingencies</i> <i>Terminology</i> <i>Costs and Benefits</i>	Describe Data <i>Data Description Report</i>	Clean Data <i>Data Cleaning Report</i>	Generate Test Design <i>Test Design</i>	Review Process <i>Review of Process</i>	Plan Monitoring and Maintenance <i>Monitoring and Maintenance Plan</i>
Determine Data Mining Goals <i>Data Mining Goals</i> <i>Data Mining Success Criteria</i>	Explore Data <i>Data Exploration Report</i>	Construct Data <i>Derived Attributes</i> <i>Generated Records</i>	Build Model <i>Parameter Settings</i> <i>Models</i> <i>Model Descriptions</i>	Determine Next Steps <i>List of Possible Actions</i> <i>Decision</i>	Produce Final Report <i>Final Report</i> <i>Final Presentation</i>
Produce Project Plan <i>Project Plan</i> <i>Initial Assessment of Tools and Techniques</i>	Verify Data Quality <i>Data Quality Report</i>	Integrate Data <i>Merged Data</i>	Assess Model <i>Model Assessment</i> <i>Revised Parameter Settings</i>		Review Project <i>Experience</i> <i>Documentation</i>
		Format Data <i>Reformatted Data</i> <i>Dataset</i> <i>Dataset Description</i>			

데이터 전처리

Data Refinement/Cleansing

- 정제되지 않은 데이터를 분석에 적합한 형태로 만드는 과정
- 데이터에 빈 부분이 있거나 무결성과 정확성이 깨져 있는 상황 파악



데이터 전처리

Data Integrity

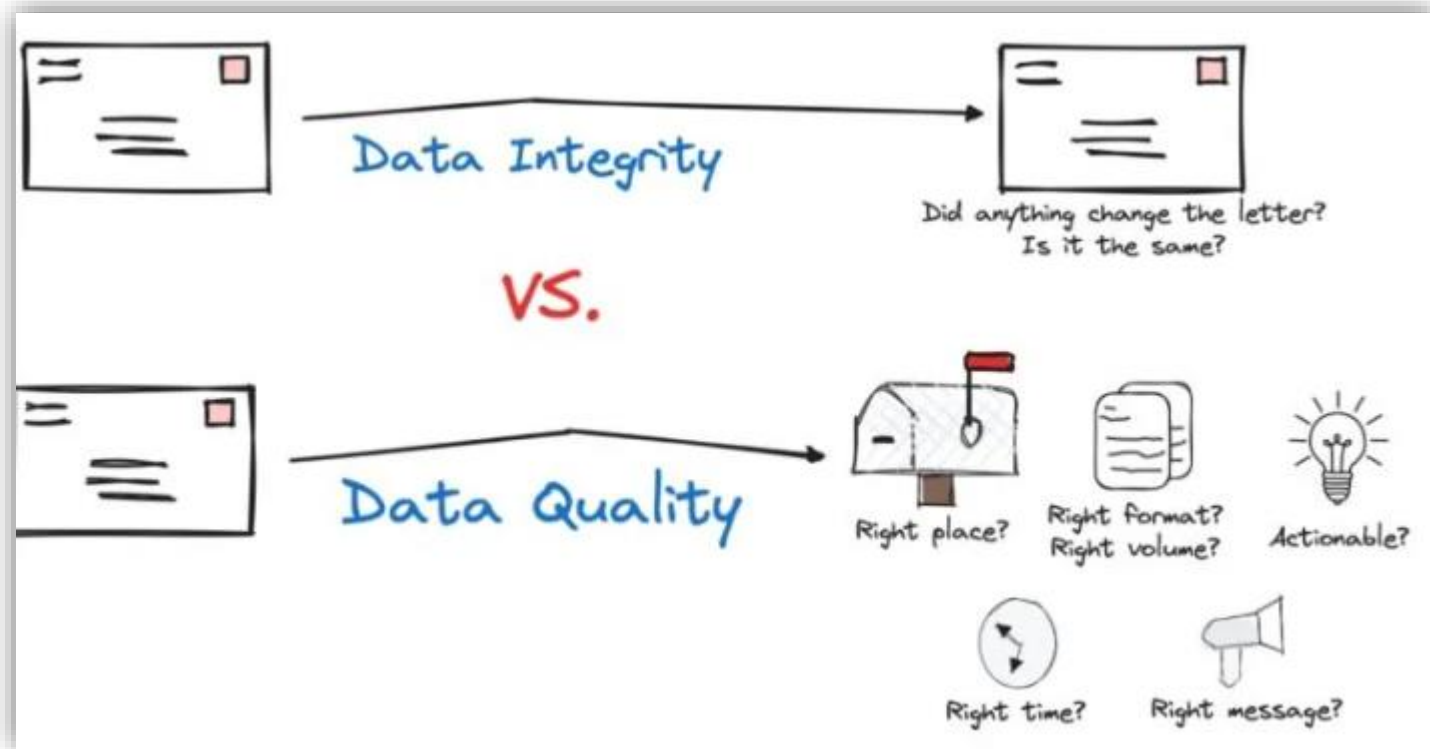
데이터 무결성 :

- 데이터 값이 정확한 상태(완전성, 정확성)
- 데이터의 정확성, 일관성, 신뢰성을 유지한 상태
- 데이터가 손상되지 않고 원래의 상태를 유지하고 있는 것.
- 데이터의 품질이 보장된 상태

ex) 시험 점수 범위가 0~100점, A학생 점수 -1점

데이터 전처리

Data Integrity



데이터 전처리

Data Integrity



Practices to preserve data integrity. Adapted from [Varonis](#)

데이터 전처리

Data Integrity



아이디

@naver.com

비밀번호

🔒

비밀번호 재확인

🔒

이름

생년월일

년(4자)

월

▼

일

성별

성별

▼

본인 확인 이메일(선택)

휴대전화

대한민국 +82

▼

데이터 전처리

Data Consistency

데이터 정합성,

- 데이터가 서로 모순되지 않고 일관되게 일치함
- 데이터의 동일성이 보장되는 상태
- 여러 시스템이나 걸쳐 동일하게 유지되는 상태

ex, 동일 데이터, 서로 모순되는 정보 -> consistency가 깨진 상태

id	이름		id	이름
1	김력스	≠	1	김로보

데이터 전처리

Data Pre-processing

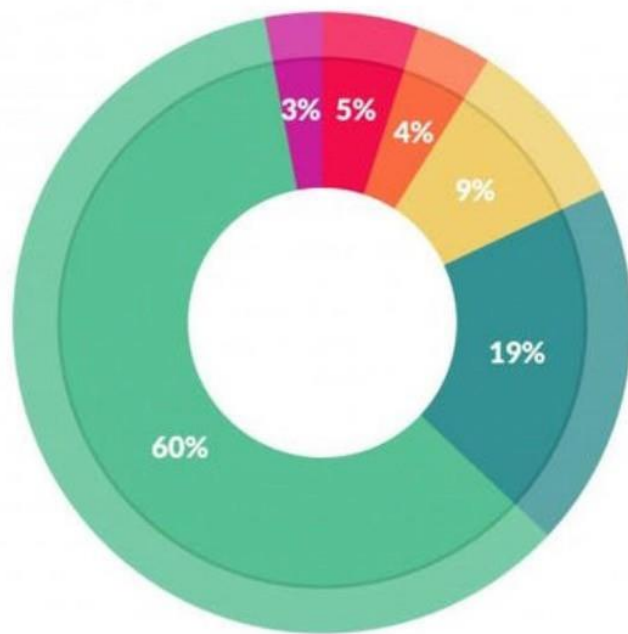
- 정제되지 않은 데이터를 분석에 적합한 형태로 만드는 과정
- 훌륭한 모델링? vs 품질 좋은 데이터?
- 데이터 분석에서 가장 중요한 과정 중 하나



데이터 전처리

Data Pre-processing

- 데이터 분석 과정 중 가장 많은 시간 소요



What data scientists spend the most time doing

- Building training sets: 3%
- Cleaning and organizing data: 60%
- Collecting data sets; 19%
- Mining data for patterns: 9%
- Refining algorithms: 4%
- Other: 5%

<https://www.forbes.com/sites/gilpress/2016/03/23/data-preparation-most-time-consuming-least-enjoyable-data-science-task-survey-says>

데이터 전처리

Data Pre-processing

▪ 데이터 전처리 작업 (기법)

- 결측치 처리: 데이터에서 빠진 값이 있을 경우, 해당 값을 대체하거나 삭제
- 이상치 처리: 데이터에서 이상 값이 있을 경우, 해당 값을 대체하거나 삭제
-> 분석 결과에 영향을 미치는 오류를 방지
- 데이터 정규화: 서로 다른 스케일의 데이터 일정한 범위로 조정
- 데이터 인코딩: 분석 효율을 위한 변환 (예: 원-핫 인코딩)
비 수치 데이터(categorical 변수 ex, 텍스트 데이터)를 컴퓨터가 이해할 수 있는 형태로 변환
- 데이터 통합: 여러 개의 데이터를 하나의 데이터로 통합 -> 분석 효율성
- 데이터 분할: 분석에 필요한 부분 데이터를 추출, 불필요한 데이터 제거
- 데이터 정렬: 분석에 필요한 순서대로 데이터 정렬

데이터 전처리

Data Pre-processing

▪ 데이터 전처리 작업 (기법)

- 결측치 처리: `df.fillna(df.mean())`
- 이상치 처리: `df=df[(df['value'] > lower_bound) & (df['value'] < upper_bound)]`
-> 특정 범위 내에 있는 값만 남기기
- 데이터 정규화: `scaler = MinMaxScaler()`
- 데이터 인코딩 `pd.get_dummies(df['category_column'])` (원-핫 인코딩)
- 데이터 통합: `df = pd.concat([df1, df2], axis=0)` (두 데이터를 행 기준으로 통합)
- 데이터 분할: `df[['column1', 'column2']]`
- 데이터 정렬: `df.sort_values(by='column_name', ascending=True)`

데이터 전처리

Data Pre-processing

▪ 데이터 전처리 작업 (기법)

- **데이터 그룹화:** 데이터를 그룹별로 분류, 각 그룹에 대한 통계 정보를 추출 (ex, 그룹별 평균, 합계, 표준편차 등)
- **데이터 변환 함수:** 데이터 값 변환 (apply, map, applymap 등 함수 사용)
- **데이터 피벗:** 행과 열을 바꾸거나, 그룹별 집계 정보 표현 (pivot 함수)
- **데이터 병합:** 여러 개의 데이터를 하나로 병합 (merge 함수)
- **데이터 분할:** 데이터 분할, 분할된 데이터 분석 (split 함수)
- **데이터 샘플링:** 샘플 데이터 추출, 추출된 데이터 분석 (sample 함수)
- **데이터 집계:** 그룹별 집계 정보 추출 (agg 함수)
- **데이터 시각화:** 데이터 시각화 (matplotlib, seaborn)

데이터 전처리

■ 실습

2.03.PreProcessing.Basic.ipynb

기본 데이터 전처리 실습

데이터 전처리

1. 데이터 구조

2. 데이터 분할

3. 결측치 처리

4. 데이터 변환

데이터 구조

데이터 전처리

데이터 분류(Data Classification) - 데이터 특성 (Attribute)

범주형 Categorical

질적 데이터
(정성적 데이터)

명목형 데이터
Nominal

성별, 혈액형 등
분류된 자료

순서형 데이터
Ordinal

만족도, grade 등
순서 관계가 존재하는 자료

수치형 Numerical

양적 데이터
(정량적 데이터)

이산형 데이터
discrete

인원, 방문 수 등
이산적인 값을 갖는 자료

연속형 데이터
continuous

신장, 체중 등
연속적인 값을 갖는 자료

데이터 전처리

데이터 구조 - 차원

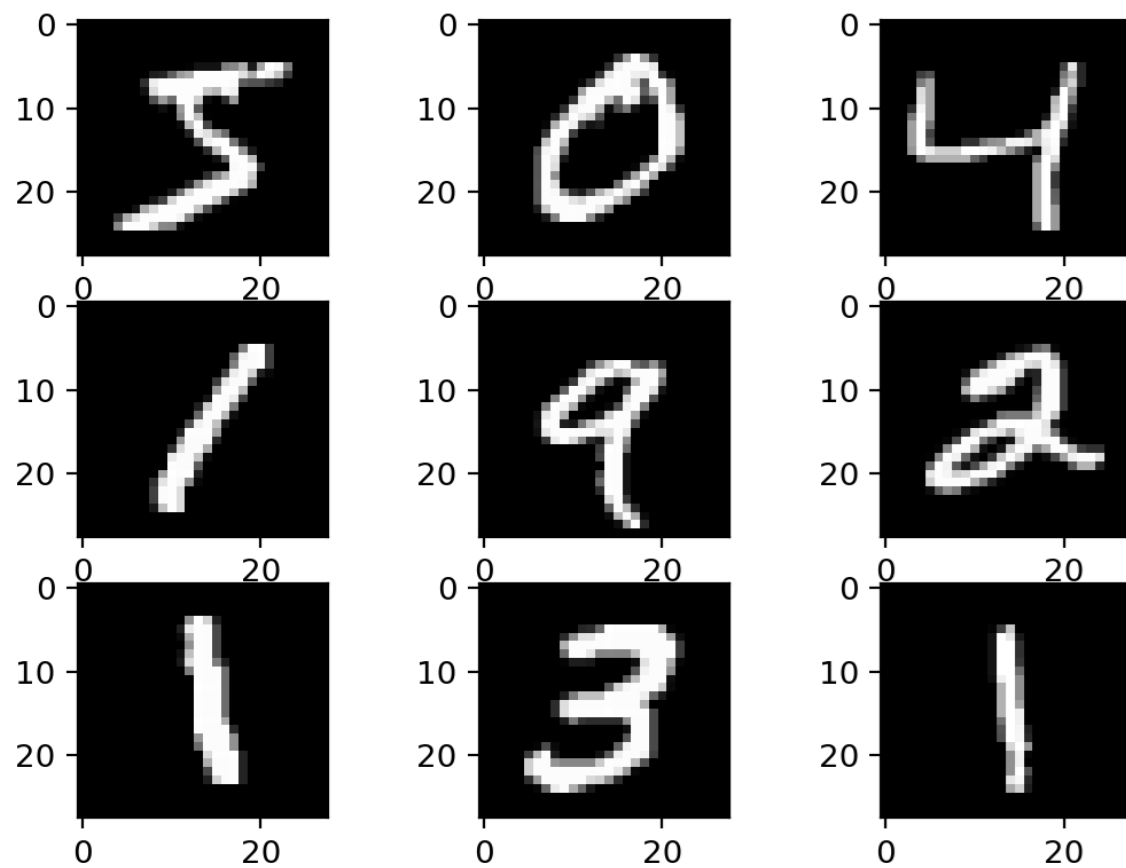
2차원, Table, Array, DataFrame

Target		Features				
Label	V01	V02	V03	V04	...	V##

데이터 전처리

데이터 구조 - 차원

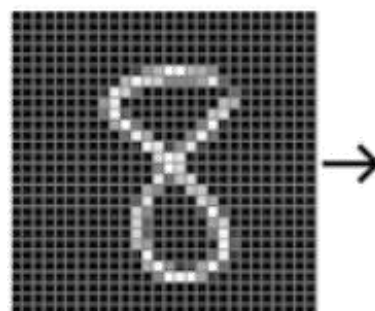
3차원, image



데이터 전처리

데이터 구조 - 차원

3차원, image



28 x 28
784 pixels

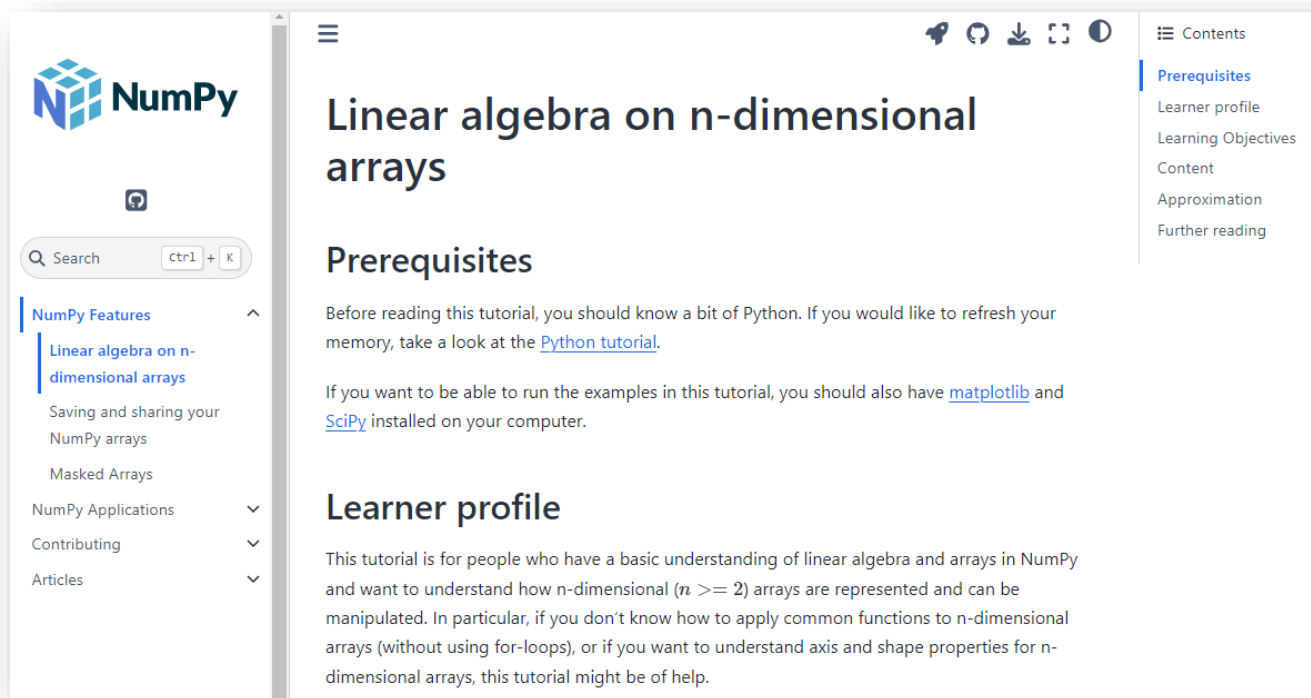
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	4	62	146	182	254	254	181	176	139	15	0	0	0	0
0	0	0	0	0	0	0	0	0	0	34	186	253	217	208	136	136	136	166	232	99	0	0	0
0	0	0	0	0	0	0	0	0	61	242	208	111	3	0	0	0	0	18	32	107	43	0	0
0	0	0	0	0	0	0	0	0	156	242	23	0	0	0	0	0	0	13	191	181	6	0	0
0	0	0	0	0	0	0	0	0	121	255	98	3	0	0	0	0	0	8	194	225	12	0	0
0	0	0	0	0	0	0	0	0	0	169	253	120	3	0	0	0	0	128	247	51	0	0	0
0	0	0	0	0	0	0	0	0	3	111	244	169	19	0	14	131	249	117	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	59	241	235	72	142	229	66	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	25	218	254	231	36	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	133	253	221	33	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	19	237	111	196	217	19	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	174	138	0	23	193	204	18	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	96	224	0	0	0	25	218	169	3	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	215	138	0	0	0	0	86	253	99	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	215	97	0	0	0	0	3	162	214	11	0	0	0
0	0	0	0	0	0	0	0	0	0	0	215	97	0	0	0	0	0	118	253	68	0	0	0
0	0	0	0	0	0	0	0	0	0	0	185	157	0	0	0	0	0	40	254	98	0	0	0
0	0	0	0	0	0	0	0	0	0	0	50	244	61	0	0	0	0	112	244	58	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	174	251	142	59	83	167	244	111	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	6	133	253	253	253	169	61	3	0	0	0	0	0

데이터 전처리

데이터 구조 - 차원

3차원, image

<https://numpy.org/numpy-tutorials/content/tutorial-svd.html>



데이터 전처리

데이터 구조 - 차원



NumPy

1D Array

3	2
---	---

[3, 2]

2D Array

1	0	1
3	4	1

[[1, 0], [3, 4]]

3D Array

1	7	9
5	9	3
7	9	9



	Name	Code-Name	Age	Weight in kgs
0	Amar	Alpha	23	69
1	Barsha	Bravo	25	54
2	Carlos	Charlie	22	73
3	Tanmay	Tango	27	70
4	Misbah	Mike	29	74

데이터 분할

데이터 전처리

데이터 전처리 - 결과물

- 결측치 없는
- 동일(유사한) 스케일의
- 수치 데이터

Target		Features				
Label	V01	V02	V03	V04	...	V##

데이터 전처리

데이터 전처리 - 분할

```
X = iris.data  
y = iris.target
```

y
Target,
Label,
Output,
종속 변수

Label	V01		V03	V04	...	V##

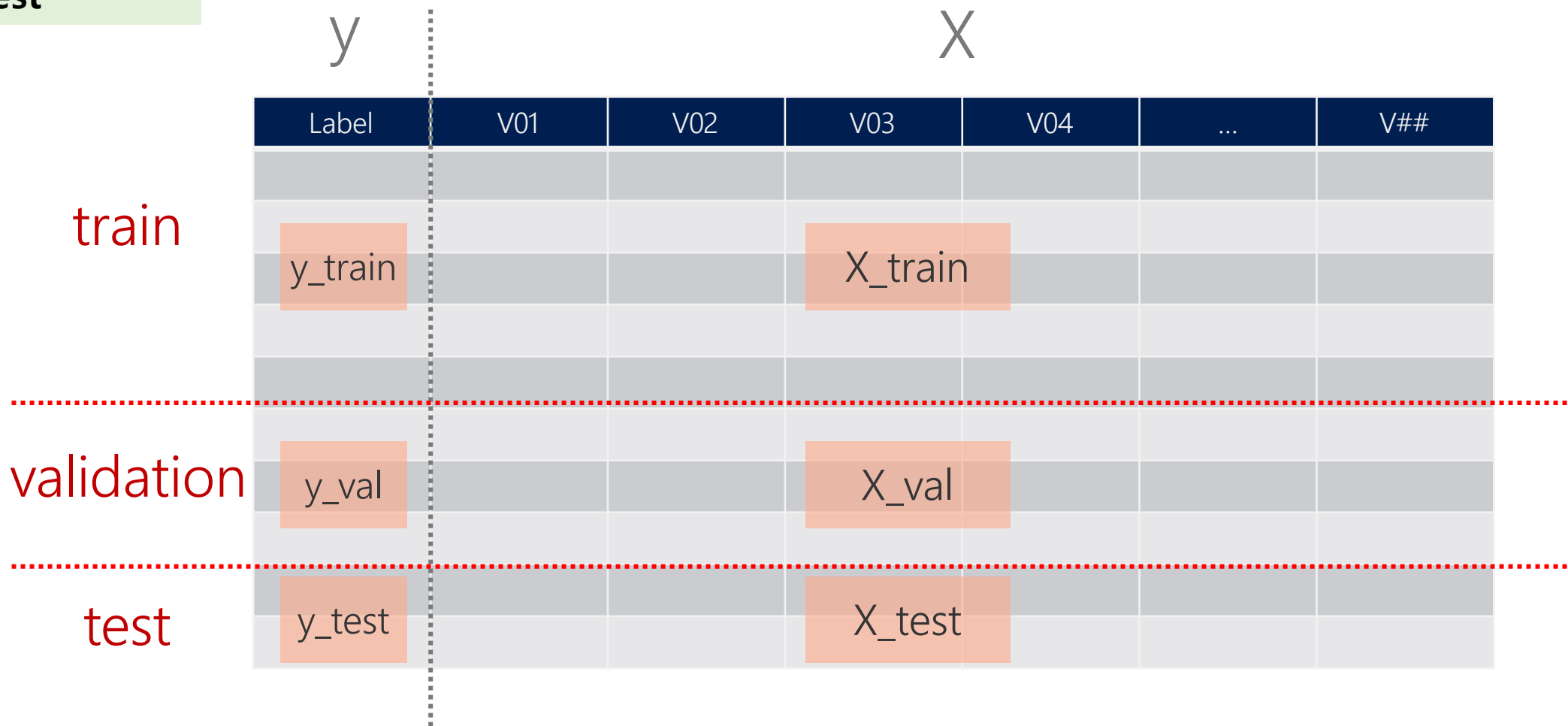
X
Feature,
columns,
input,
요인,
독립 변수

데이터 전처리

데이터 전처리 - 분할

- train
- validation
- test

```
X_train, X_test, y_train, y_test  
= train_test_split(X, y, test_size=0.3, random_state=42)
```



데이터 전처리

데이터 전처리 - 분할

- train
- validation
- test

훈련 데이터 정확도: 1.0000
검증 데이터 정확도: 0.9545

```
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
```

```
# 데이터 분할 (Train 70%, Validation 15%, Test 15%)
```

```
X_train, X_temp, y_train, y_temp = train_test_split(X, y, test_size=0.3, random_state=42)
```

```
X_val, X_test, y_val, y_test = train_test_split(X_temp, y_temp, test_size=0.5, random_state=42)
```

```
# 모델 학습
```

```
model = RandomForestClassifier(n_estimators=10, max_depth=10, random_state=42)
model.fit(X_train, y_train)
```

```
# 성능 평가
```

```
train_acc = accuracy_score(y_train, model.predict(X_train))
val_acc = accuracy_score(y_val, model.predict(X_val))
```

```
print(f"훈련 데이터 정확도: {train_acc:.4f}")
```

```
print(f"검증 데이터 정확도: {val_acc:.4f}")
```

데이터 전처리

데이터 전처리 - 분할

- **Train**

- 모델을 **학습**시키는 데 사용되는 데이터.
- 모델이 입력 데이터와 그에 상응하는 출력(레이블)을 학습하여 패턴을 찾아내는 과정에서 사용
- (보통) 전체 데이터의 60-80%

- **Validation**

- 모델 성능 평가 -> 모델 하이퍼파라미터 조정, 오버피팅(overfitting) 방지
- 훈련 세트로 학습 후 **검증 세트에서 얼마나 잘 예측하는지 평가** -> 모델의 성능 조정
- 훈련에 사용된 데이터와는 다른 데이터로 성능 측정 -> 오버피팅 방지
- (보통) 전체 데이터 10-20%

- **Test**

- 최종적인 모델 성능 평가
- 훈련이나 검증 과정에서 사용하지 않은 데이터 사용 -> 모델의 일반화(generalize) 성능 평가
- (보통) 전체 데이터의 10-20%

데이터 전처리

데이터 전처리 - 분할

- Train
 - 모델을 **학습**시키는 데 사용되는 데이터 (학습 : 모델 파라미터 업데이트)
- Validation
 - 모델 성능 평가 -> 모델 하이퍼파라미터를 조정, 오버피팅(overfitting) 방지

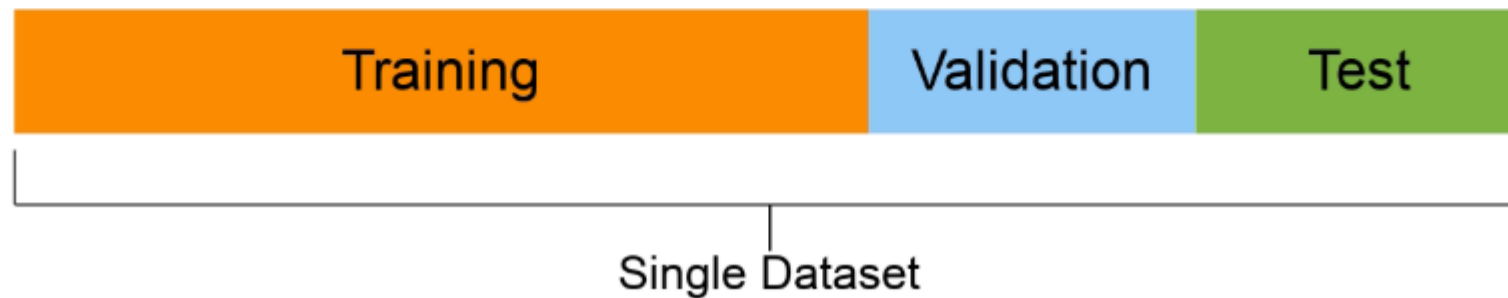


- 모델 파라미터 : 모델이 학습하는 과정에서 데이터로부터 자동으로 결정되는 값
ex, 선형 회귀에서의 가중치(weight)
- 하이퍼파라미터 : ML 모델 학습 과정에서 분석가가 직접 설정하는 모델 설정을 위한 파라미터

데이터 전처리

데이터 전처리 - 분할

- train
- validation
- test



데이터 전처리

데이터 전처리 - 분할

- **train**
- **validation**
- **test**

```
import pandas as pd
from sklearn.model_selection import train_test_split

# Split the data into features and target variable
# Assuming 'target' is the name of the target variable
X = data.drop('target', axis=1) # Creditability
y = data['target'] # Creditability

# Split the data into training and testing sets (e.g., 80% train, 20% test)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Further split the training set into training and validation sets (e.g., 75% train, 25% validation)
X_train, X_val, y_train, y_val = train_test_split(X_train, y_train, test_size=0.25, random_state=42)

# 60% training, 20% validation, and 20% test sets
```

데이터 전처리

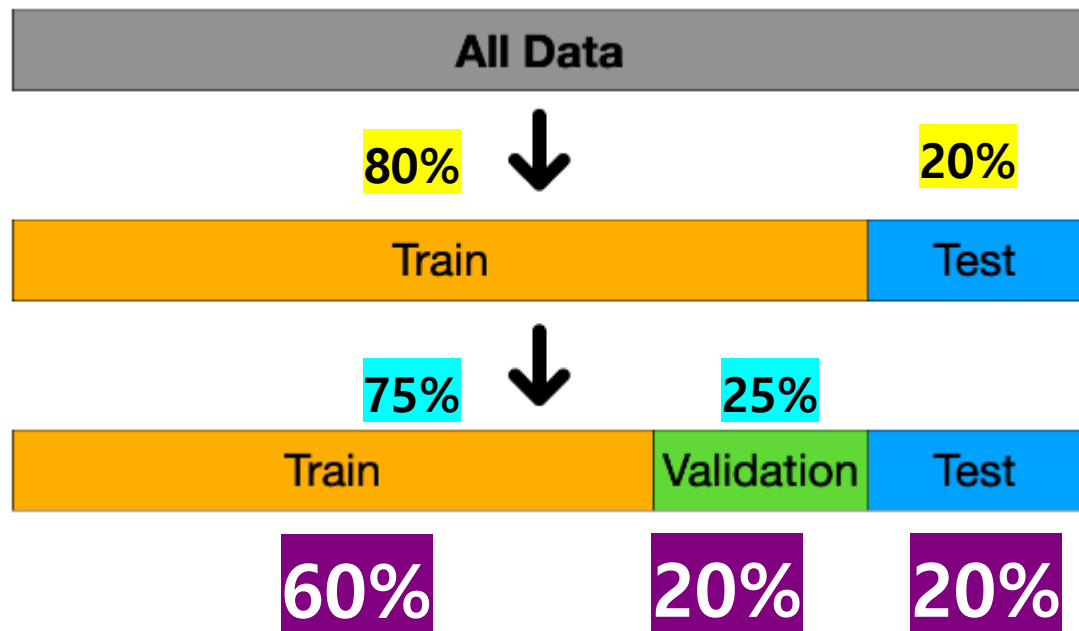
- train
- validation
- test

데이터 전처리 - 분할

```
# Split the data into training and testing sets (e.g., 80% train, 20% test)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Further split the training set into training and validation sets (e.g., 75% train, 25% validation)
X_train, X_val, y_train, y_val = train_test_split(X_train, y_train, test_size=0.25, random_state=42)

# 60% training, 20% validation, and 20% test sets
```



60%는 훈련 데이터,
20%는 검증 데이터,
20%는 테스트 데이터로 나눔

데이터 전처리

데이터 전처리 - 분할 : Cross-Validation

교차 검증

머신 러닝 모델의 성능을 보다 안정적이고 신뢰성 있게 평가하기 위해 데이터를 여러 번 나누어 학습과 검증을 반복하는 방법

4-fold validation (k=4)



데이터 전처리

데이터 전처리 – 분할 : Cross-Validation

교차 검증

머신 러닝 모델의 성능을 보다 안정적이고 신뢰성 있게 평가하기 위해 데이터를 여러 번 나누어 학습과 검증을 반복하는 방법

- 실습

2.03.PreProcessing.Basic.ipynb

- Cross-Validation

데이터 전처리

데이터 전처리 - 분할

- 모델 파라미터 : 모델이 학습하는 과정에서 데이터로부터 자동으로 결정되는 값
ex, 선형 회귀에서의 가중치(weight)
- 하이퍼파라미터 : ML 모델 학습 과정에서 분석가가 직접 설정하는 모델 설정을 위한 파라미터

2 HyperParameter.pdf

결측치, 이상치 처리

데이터 전처리

■ 실습

2.03.Data_ Pre.processing.ipynb

- 타이타닉 데이터 셋

	passengerId	survived	pclass	name	sex	age	sibsp	parch	ticket	fare	cabin	embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S
5	6	0	3	Moran, Mr. James	male	NaN	0	0	330877	8.4583	NaN	Q
6	7	0	1	McCarthy, Mr. Timothy J	male	54.0	0	0	17463	51.8625	E46	S
7	8	0	3	Palsson, Master. Gosta Leonard	male	2.0	3	1	349909	21.0750	NaN	S

데이터 전처리

실습 : 타이타닉 데이터셋 - 결측치

■ 결측치

- 데이터에 값이 없는 것
- 데이터를 수집하는 과정에서 오류나 실수로 발생
- 처리 :
 - 1) 다른 값으로 대체 (ex, 0 또는 평균값)
 - 2) 결측치 제거

■ 데이터프레임에서 결측치 확인

- info 함수
 - Non-Null : 결측치가 아닌 값 (나머지는 결측치)
 - object는 문자형, int64는 정수형, float64는 실수형

```
Data columns (total 12 columns):
#   Column      Non-Null Count  Dtype
---  -
0   PassengerId  156 non-null    int64
1   Survived     156 non-null    int64
2   Pclass       156 non-null    int64
3   Name         156 non-null    object
4   Sex          156 non-null    object
5   Age          126 non-null    float64
6   SibSp        156 non-null    int64
7   Parch        156 non-null    int64
8   Ticket       156 non-null    object
9   Fare         156 non-null    float64
10  Cabin        31 non-null     object
11  Embarked     155 non-null    object
dtypes: float64(2), int64(5), object(5)
```

데이터 전처리

실습 : 타이타닉 데이터셋 - 결측치

■ isna 함수

- 결측치가 있는지 확인, 결측치가 있으면 NaN 반환
- NaN(Not a Number) : 판다스의 결측치 표기

```
1 titanic_df.isna()
```

	passengerId	survived	pclass	name	sex	age	sibsp	parch	ticket	fare	cabin	embarked
0	False	False	False	False	False	False	False	False	False	False	True	False
1	False	False	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False	True	False
3	False	False	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False	True	False
...
151	False	False	False	False	False	False	False	False	False	False	False	False
152	False	False	False	False	False	False	False	False	False	False	True	False
153	False	False	False	False	False	False	False	False	False	False	True	False

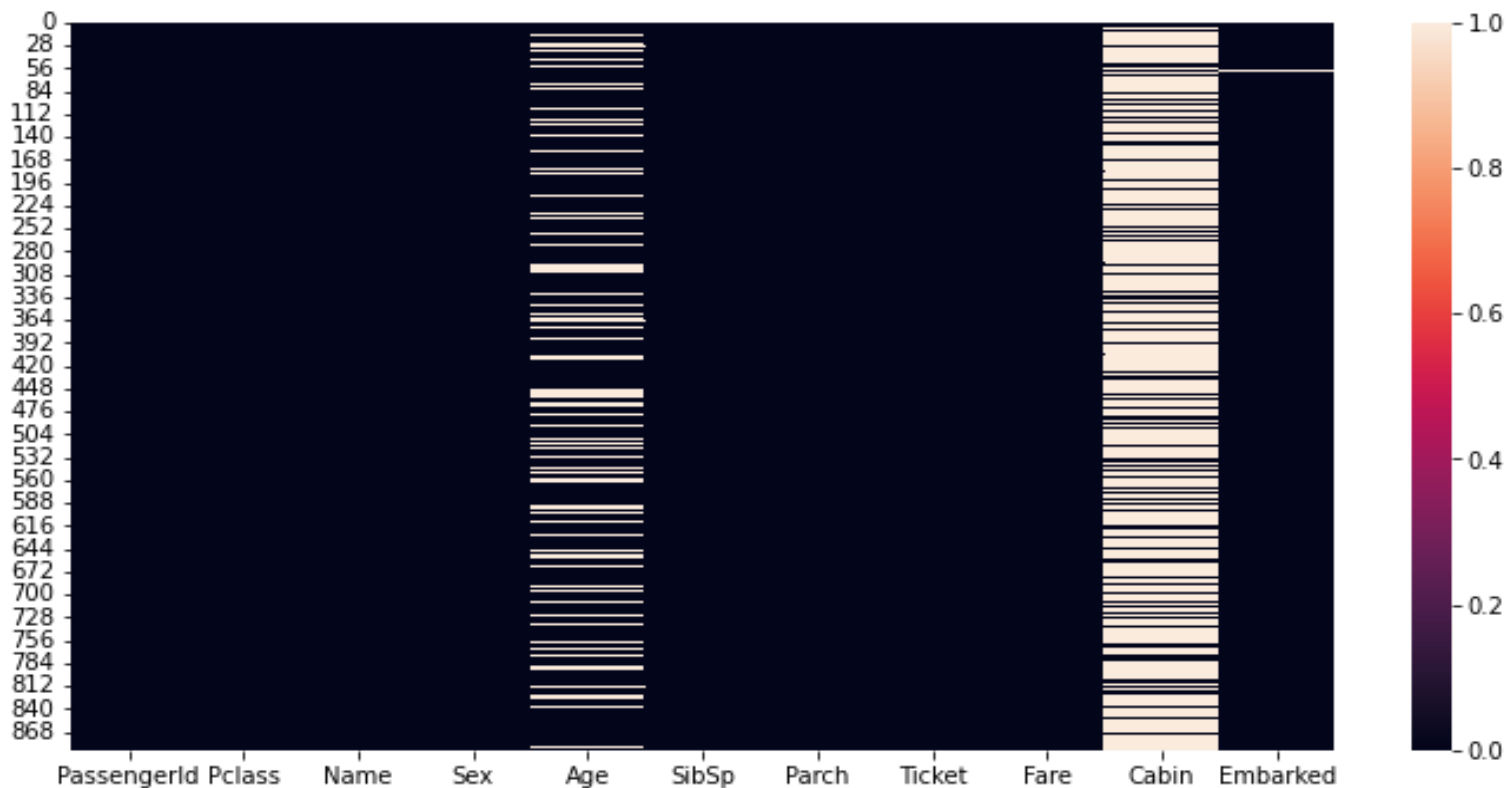
데이터 전처리

실습 : 타이타닉 데이터셋 - 결측치

- 결측치 시각화

isna() + sns.heatmap

```
import seaborn as sns
sns.heatmap(titanic_df.isna())
```



데이터 전처리

실습 : 타이타닉 데이터셋 - 결측치

■ isnull 함수

- 결측치가 있는지 확인 -> 결측치가 있으면 불 인덱스 True 반환
- with sum 함수 -> NaN 값 총 갯수 확인

■ notnull 함수

- 결측치가 있는지 확인 -> 결측치가 없으면 불 인덱스 True 반환
- cf, isnull 함수

`'isna()' = 'isnull()'`

데이터 전처리

실습 : 타이타닉 데이터셋 - 결측치

- 결측치 처리

- 1) 다른 값으로 대체 (ex, 0 또는 평균값)

- 다른 값으로 대체 (ex, 0 또는 평균값)

- fillna 함수 사용 결측치 값 변경
 - count 함수 사용, 특정 열에서 개별 값 count 확인 -> 결측치 처리 전후 비교

- 결측치를 평균값으로 변경

- mean 함수, 특정 열의 평균값 구함
 - fillna 함수, 해당 열의 평균값으로 변경 가능
 - 결측치인 경우에만 평균값으로 변경되고, 결측치가 아니면 원래 값 표시

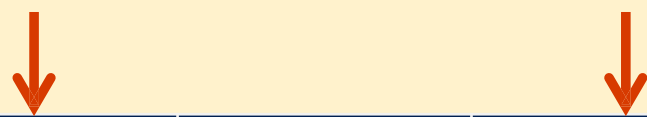
데이터 전처리

실습 : 타이타닉 데이터셋 - 결측치

- 결측치 처리

- 2) 결측치 제거

1) NA가 있는 열 제거



Two orange arrows point down to the x01 and x03 columns of the table, indicating they are the target for removal due to missing values.

y	x01	x02	x03	x04
1	@@@	2	.45	AAA
0	###	3	.9	XXX
1	&&&	6	0	BBB
0	%%%	23	<NA>	AAA
1	!!!	2	.75	AAA
0	<NA>	32	.21	XXX

2) NA가 있는 행 제거



데이터 전처리

실습 : 타이타닉 데이터셋 - 결측치

- 결측치 처리

2) 결측치 제거 - `dropna()` 메소드, 결측치가 있는 열 or 행 제거

- 어떤 열이든 결측치가 있는 행 제거

```
# 결측치가 하나라도 있는 행 제거  
titanic_df.dropna(axis=0, inplace=True)
```

- 특정 열에 결측치가 있는 행 제거

```
# Ozone 열이 결측치인 행 제거  
titanic_df.dropna(subset=['age'], axis=0, inplace=True)
```

- 결측치가 있는 열 제거

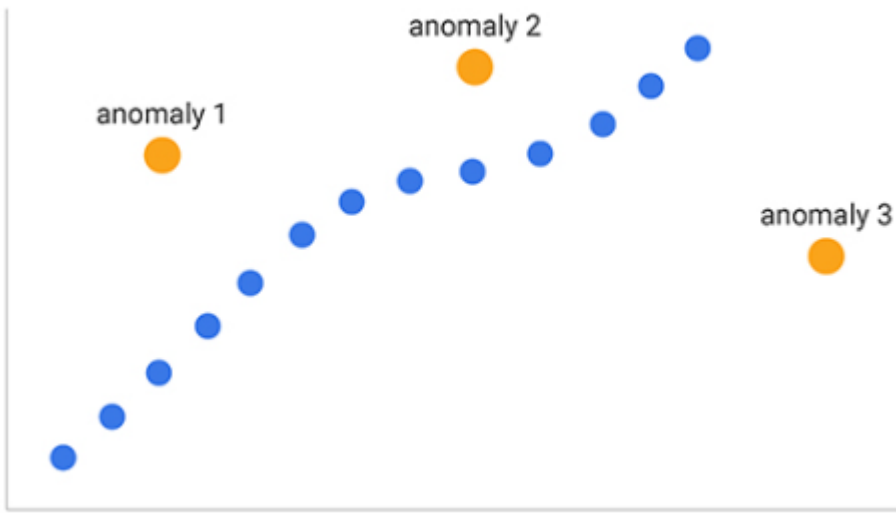
```
# 결측치가 있는 열 제거  
titanic_df.dropna(axis=1, inplace=True)
```

데이터 전처리

데이터 전처리 – 이상치 (Anomaly)

- 이상치

- 데이터의 범위에서 많이 벗어난 값
ex, 사람 체온 20도 or 50도
- 데이터 수집 과정에 동반되는 오류
- (결측치와 같이) 처리(ex, 제거 등) 필요



데이터 전처리

데이터 전처리 – 이상치 (Anomaly)

- 이상치 확인

- df.value_counts() 메서드(빈도 출력) -> sort_index 정렬
=> 이상치 확인

```
print(middle_exam['id'].value_counts().sort_index())  
print(middle_exam['score'].value_counts().sort_index())
```

2016120033	1
2018333333	1
2019230034	1
2020110017	1
2020330012	1
2022210067	1
3022130006	1



3022년도 신입생 ?

10.0	1
15.0	1
18.0	1
20.0	1
22.0	1
25.0	1
330.0	1



330점 ?

데이터 전처리

데이터 전처리 – 이상치 (Anomaly)

- 결측치로 대체

- numpy where 메서드 이용, score 값 중 정해진 범위를 벗어난 값 확인 -> score 값을 결측값(Nan) 대체

```
middle_exam['score'] = np.where(middle_exam['score'] > 30, np.nan, middle_exam['score'])  
print(middle_exam)
```

	id	score
0	2020110017	25.0
1	NaN	20.0
2	2018333333	NaN
3	2016120033	10.0
4	2019230034	15.0
5	2020330012	22.0
6	2022210067	18.0
7	3022130006	NaN

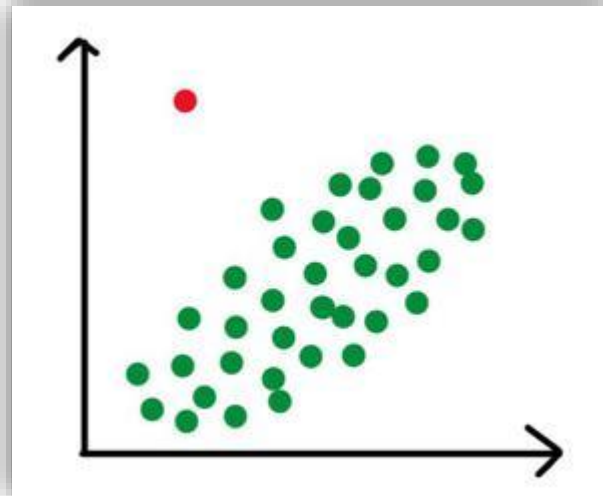
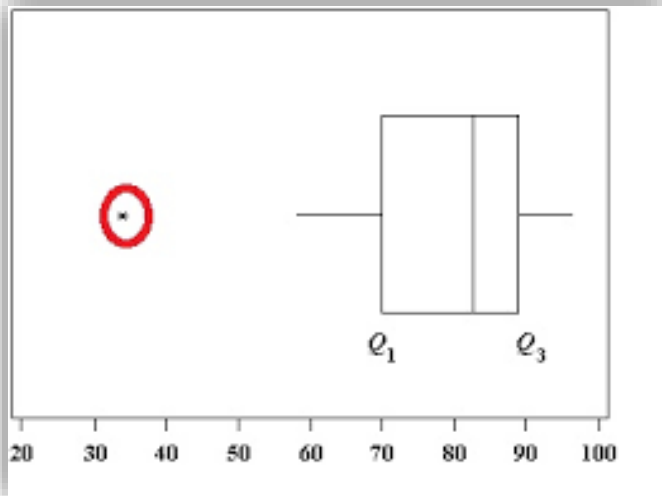


NaN 변환

데이터 전처리

데이터 전처리 - 극단치 (Outlier)

- 극단치
 - 극단적으로 크거나 작은 값(존재할 확률이 매우 낮은 경우)
 - cf, 이상치 : 논리적으로 있을 수 없는 값
 - ex, 키 2m30cm
 - 분석, 모델링에 영향(어려움) -> 제거



데이터 전처리

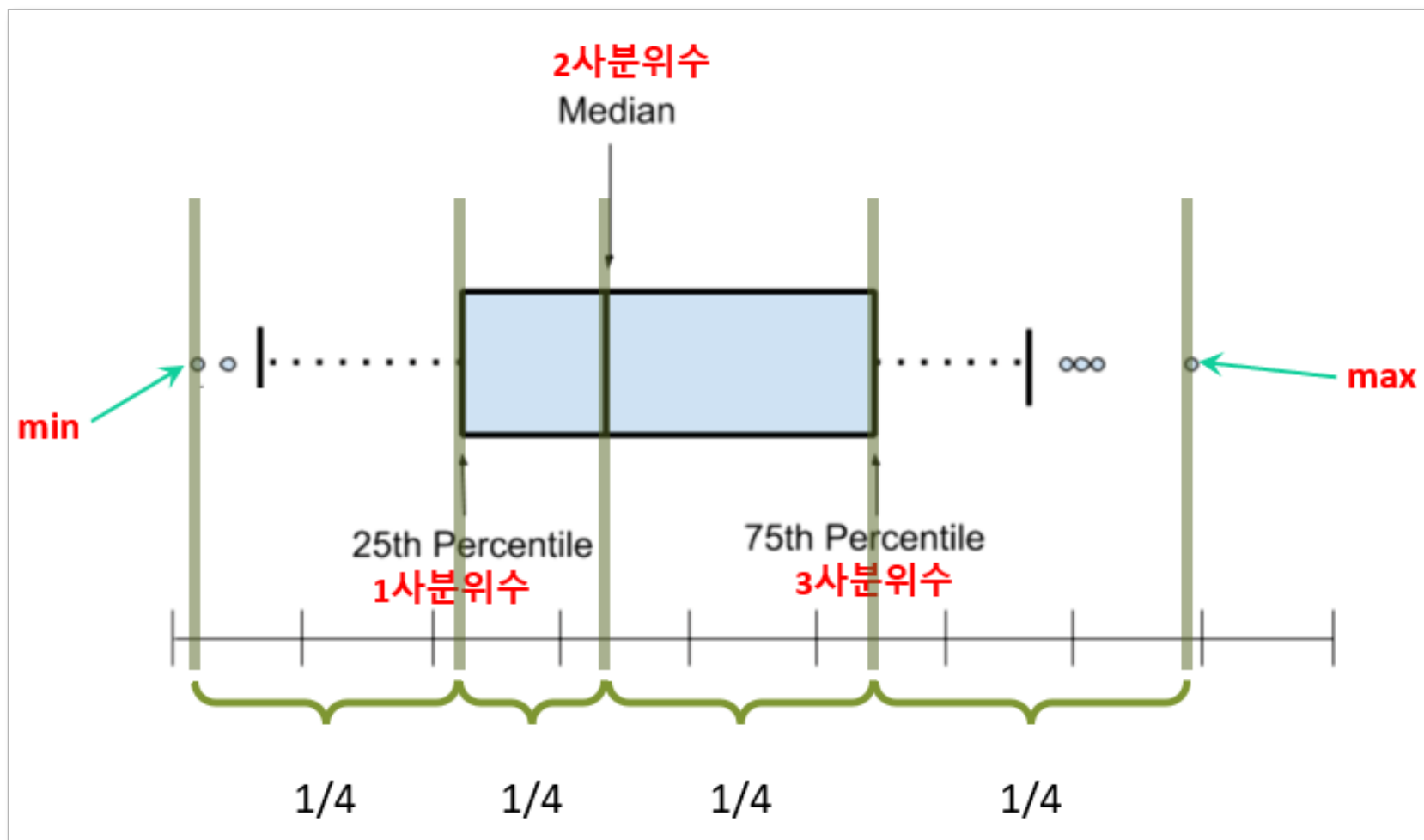
데이터 전처리 – 극단치 (Outlier)

극단치 기준?
정상 범주 기준?

데이터 전처리

데이터 전처리 - 극단치 (Outlier)

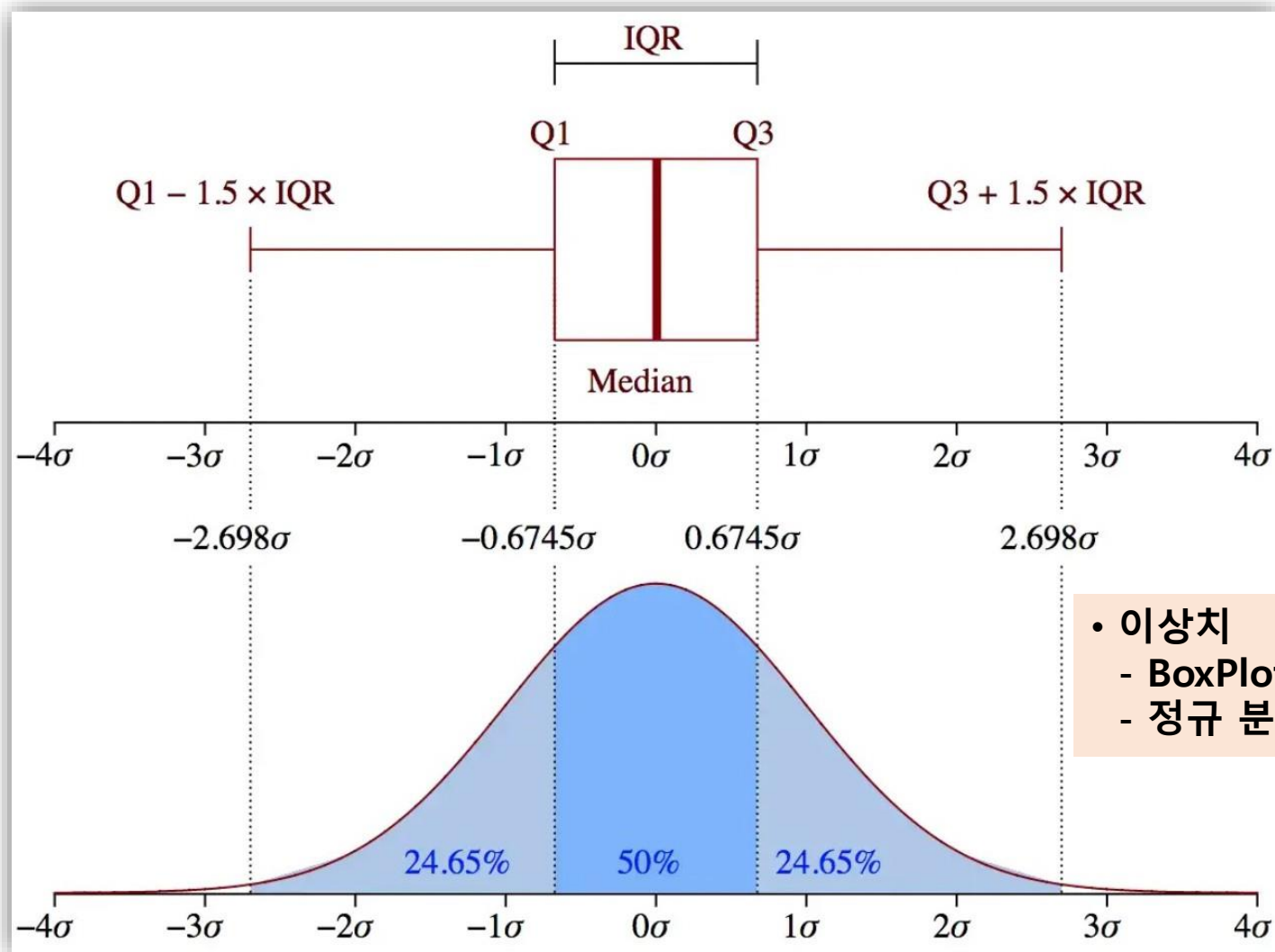
■ **Matplotlib - boxplot** 데이터의 분포를 시각적으로 표현, 중앙값과 사분위수 확인



데이터 전처리

데이터 전처리 - 극단치 (Outlier)

■ IQR with Boxplot



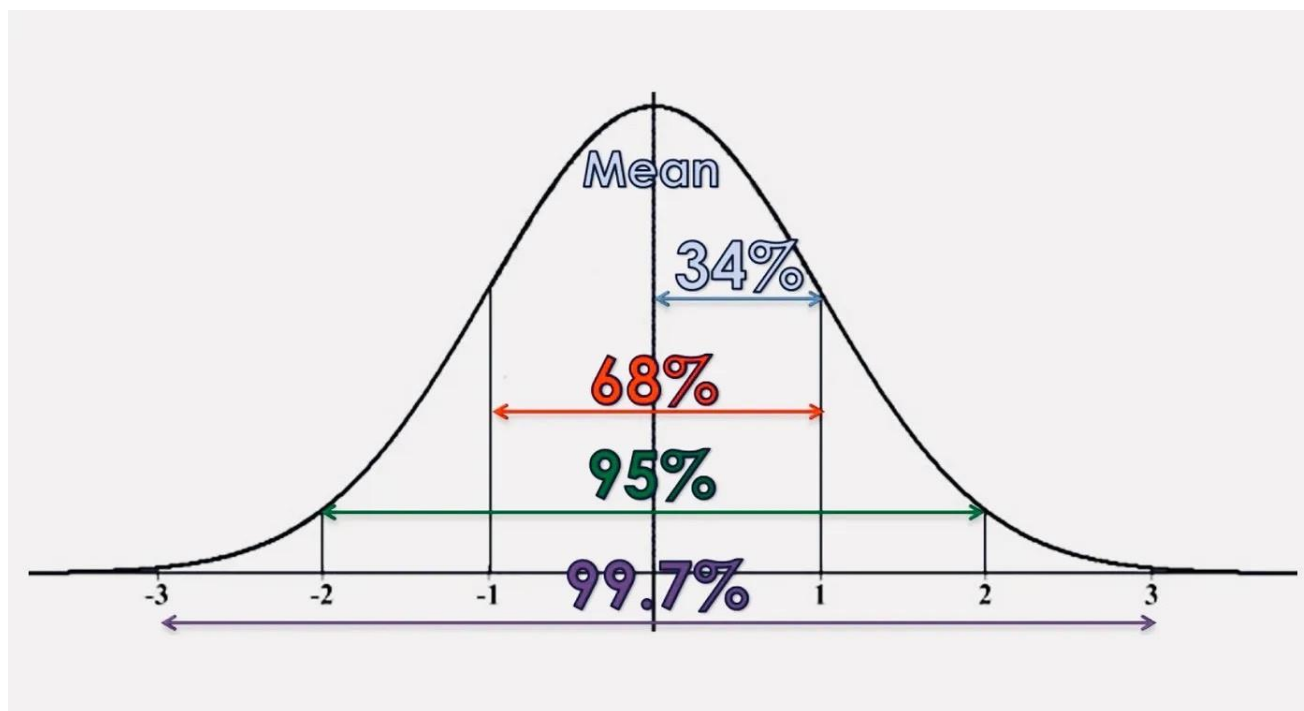
• 이상치

- BoxPlot : $Q1 - 1.5 \times IQR$ 이하 or $Q3 + 1.5 \times IQR$ 이상
- 정규 분포 : -2.698σ 이하 or 2.698σ 이상의 데이터

σ : 표준 편차(Standard Deviation)
평균으로부터 떨어진 정도

데이터 전처리

표준편차 정규분포



데이터 전처리

데이터 전처리 – 극단치(Outlier) 처리 w.IQR

```
import pandas as pd
import numpy as np

# 샘플 데이터 생성 (정상적인 값 + 이상치 포함)
data = {'values': [10, 12, 14, 15, 17, 19, 21, 23, 100, 102]} # 이상치 : 100, 102
df = pd.DataFrame(data)

# 1사분위수 (Q1), 3사분위수 (Q3) 계산
Q1 = df['values'].quantile(0.25) # 1사분위수
Q3 = df['values'].quantile(0.75) # 3사분위수
IQR = Q3 - Q1 # IQR 계산

# IQR 이용, 이상치 하한 및 상한 계산
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

# 이상치 탐지
outliers = df[(df['values'] < lower_bound) | (df['values'] > upper_bound)]
print("이상치 데이터:\n", outliers)

# 이상치 제거 데이터
cleaned_df = df[(df['values'] >= lower_bound) & (df['values'] <= upper_bound)]
print("\n이상치 제거 후 데이터:\n", cleaned_df)
```

데이터 변환

데이터 전처리

실습 : 타이타닉 데이터셋 - 데이터 변환

■ to_numeric

- 데이터형을 숫자형
- to_numeric 함수 errors 옵션 사용
 - errors='ignore': 숫자형으로 변경할 수 없는 데이터, 원본 데이터 그대로 반환
 - **errors='coerce'**: 숫자형으로 변경할 수 없는 데이터,
기존 데이터는 지우고 NaN으로 설정하여 반환
 - errors='raise': 숫자형으로 변경할 수 없는 데이터, 오류 발생으로 코드 중단
- dtypes을 통해 숫자형 변경 확인

	survived	pclass	name	sex	age	sibsp	parch	fare
0	0	3	NaN	NaN	22.0	1	0	7.2500
1	1	1	NaN	NaN	38.0	1	0	71.2833
2	1	3	NaN	NaN	26.0	0	0	7.9250

■ astype 함수

- 원하는 데이터형으로 변경
데이터프레임 이름[열 이름].astype('변경할 데이터형')

	0
survived	float64
pclass	float64
name	float64

데이터 전처리


실습 : 타이타닉 데이터셋 - 데이터 변환

■ 숫자형 변수 변환

- 숫자형 변환 -> 특정 열의 값의 데이터 분석(수치 분석) 가능

ex, titanic_df1에 gender 열을 추가, female은 0, male은 1로 표시

- sex 열을 삭제 후 데이터 확인 (head 함수) -> 숫자로 변경 확인
- info 함수, gender열 확인



sex
male
female
female
female
male

gender
1
0
0
0
1

#	Column	Non-Null	Count	Dtype
0	survived	156	non-null	int64
1	pclass	156	non-null	int64
2	name	156	non-null	object
3	age	126	non-null	float64
4	sibsp	156	non-null	int64
5	parch	156	non-null	int64
6	fare	156	non-null	float64
7	gender	156	non-null	int64

데이터 전처리

실습 : 타이타닉 데이터셋 – 데이터 변환

■ 문자형 변수를 명목 변수로 변환

ex, 승객의 호칭 이용, 4개의 그룹 만들기

- ① 특별한 지위(Master, Don, Rev)를 나타내는 호칭을 **Special**로 변환,
-> Mr, Mrs, Miss, Special로 4개 값을 갖는 title 열 생성
 - unique와 len 함수 사용 title 열 값 확인
 - 사용하지 않는 name 열 삭제
- ② 호칭 중 Special은 1, 나머지는 0으로 바꾼 뒤 이 special_title로 열 이름 네이밍
- ③ title 열 삭제

title
Special
Special
Mrs
Mr
Mr

■ 열 데이터를 이용해 열 추가

- 두 열의 값을 합쳐서 새로운 열로 표현 (파생변수)

- **Scaling**

2.03.스케일링.pdf

데이터 전처리

- 실습

- 2.03.Data.Pre_processing.titanic.ipynb

- 타이타닉 생존율 예측

- 1. EDA

- 2. 전처리

데이터 전처리

- 실습

2.03.Data.Pre_processing.wine.ipynb

데이터 전처리

[TASK]

2.03.Data.Pre_processing.TASK.ipynb

Credit 데이터 전처리

칼럼명	설명	값 설명
Creditability	Creditability	신용도 0 : 낮은 신용도, 1 : 높은 신용도, Target
AccountBalance	Account Balance	은행잔고 1: No account, 2 : None (No balance), 3 : Some Balance
CreditDuration	Duration of Credit (month)	신청한 대출기간(월) 숫자
Payment	Payment Status of Previous Credit	과거 대출 납입 상태 0 : Delayed, 1 : Other Credits, 2 : Paid Up, 3 : No Problem with Current Credits, 4 : Previous Credits Paid
CreditAmount	Credit Amount(\$)	신청한 대출금액 숫자
Employment	Length of current employment(Month)	현 직업 근무 기간 1: Unemployed, 2: <1 Year, 3: [1, 4), 4: [4, 7), 5: Above 7
SexMarital	Sex & Marital Status	성별 & 결혼상태 1: Male, Divorced, 2: Male, Single, 3: Male, Married/Widowed, 4: Female
CurrentAddress	Duration in Current address	현 거주지 거주기간 1: <1 Year , 2: [1, 4), 3: [4, 7), 4: Above 7
Age	Age (years)	나이 숫자
AppartmentType	Type of apartment	주거환경 1: free apartment, 2: Rented, 3: Owned
Occupation	Occupation	직업 1: Unemployed, unskilled, 2: Unskilled Permanent Resident, 3: Skilled, 4: Executive
Telephone	Telephone	전화기 소유 여부 1: No, 2: Yes
ForeignWorker	Foreign Worker	외국인 근로자 여부 1: No, 2: Yes

THANK YOU