2024.10.8.

박정빈 임재성 진태완

KUGGLE 24-2
WEEK 5

피쳐 엔지니어링

FEATURE ENGINEERING

CONTENTS

CONTENTS

01

피쳐 엔지니어링이란?

데이터에서 유용한 특성을 추출하고 변형하여

머신러닝 모델의 성능을 향상시키는 과정

01

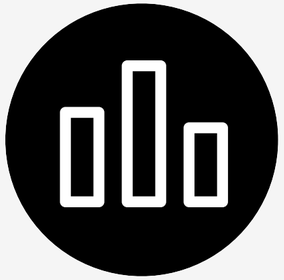
피쳐 엔지니어링이란?

데이터에서 **유용**한 특성을 **추출**하고 **변형**하여

머신러닝 모델의 **성능을 향상**시키는 과정

01

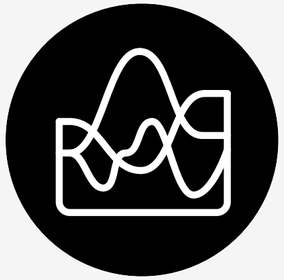
피쳐 엔지니어링이란?



데이터 전처리

변수 선택
차원 축소

파생변수



스케일링

CONTENTS

02 데이터 전처리

02

데이터 전처리

결측치

이상치

데이터 형식

02

데이터 전처리

결측치 처리 방법

1. 삭제

결측치가 포함된 행 또는 열을 제거

2. 대체

평균 / 중앙값 등으로 대체

데이터로 회귀분석/KNN을 사용하여 대체 값을 찾아서 사용

02

데이터 전처리

이상치 처리 방법

1. IQR

사분위수 / 박스 플롯을 이용해 이상치 범위 파악

2. z-score

데이터 표준화 후, 3 시그마를 넘는 데이터를 이상치로 간주

3. 도메인 지식

데이터의 분야에 적용되는 이상치 범위를 사용

02

데이터 전처리

데이터 형식 변환 - 범주형

ex) 문자열을 날짜 형식으로 변환 / 정수형 데이터를 실수로 변환

1. 원-핫 인코딩 각 범주를 새로운 이진 변수로 변환

2. 레이블 인코딩 각 범주에 고유한 정수를 부여하여 변환

CONTENTS

03 변수 선택 & 차원 축소

03

변수 선택 & 차원 축소

변수 중요도

모델 활용

차원 축소

03

변수 선택 & 차원 축소

필요성

1. 과적합 방지

2. 해석 가능성 향상

3. 계산 비용 절감

03

변수 선택

Filter Methods

1. 상관계수

```
import pandas as pd
correlation_matrix = pd.DataFrame(X).corrwith(pd.Series(y)) # X와 y의 상관계수 계산
```

2. 카이제곱 검정

```
from sklearn.feature_selection import SelectKBest, chi2
X_new = SelectKBest(chi2, k=5).fit_transform(X, y) # X는 피쳐, y는 타겟 변수
```

3. ANOVA

```
from sklearn.feature_selection import SelectKBest, f_classif
X_new = SelectKBest(f_classif, k=5).fit_transform(X, y) # X는 피쳐, y는 타겟 변수
```

03

변수 선택

Wrapper Methods

1. Foward Selection

```
from mlxtend.feature_selection import SequentialFeatureSelector
from sklearn.linear_model import LinearRegression

sfs = SequentialFeatureSelector(LinearRegression(), k_features=5, forward=True).fit(X, y) # k_features는 선택할 피쳐 수
```

2. Backward Elimination

```
sfs = SequentialFeatureSelector(LinearRegression(), k_features=5, forward=False).fit(X, y) # k_features는 선택할 피쳐 수
```

03

변수 선택

Embedded Methods

1. LASSO 회귀

```
from sklearn.linear_model import Lasso

lasso = Lasso(alpha=0.01).fit(X, y) # alpha 값은 정규화 강도
selected_features = [i for i in range(len(lasso.coef_)) if lasso.coef_[i] != 0] # 선택된 피쳐 인덱스
```

2. Random Forest

```
from sklearn.ensemble import RandomForestClassifier

rf = RandomForestClassifier().fit(X, y)
importances = rf.feature_importances_ # 피쳐 중요도 추출
```

03

변수 선택

Machine Learning

1. Recursive Feature Elimination

```
from sklearn.feature_selection import RFE
from sklearn.linear_model import LogisticRegression

selector = RFE(LogisticRegression(), n_features_to_select=5).fit(X, y) # n_features_to_select는 선택할 피쳐 수
```

2. Feature Importance from Tree-Based Models

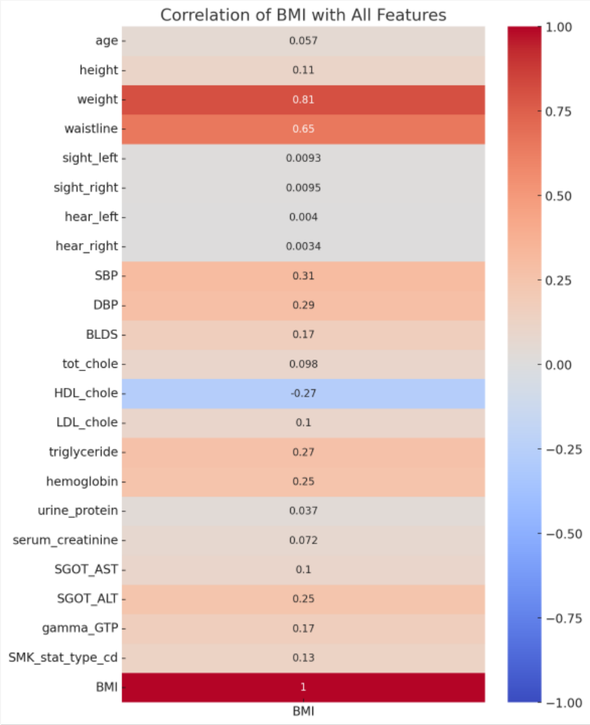
```
from sklearn.ensemble import RandomForestClassifier

rf = RandomForestClassifier().fit(X, y)
important_features = [i for i in range(len(rf.feature_importances_)) if rf.feature_importances_[i] > 0.1] # 중요도가 0.1 이상인 피쳐 인덱스
```

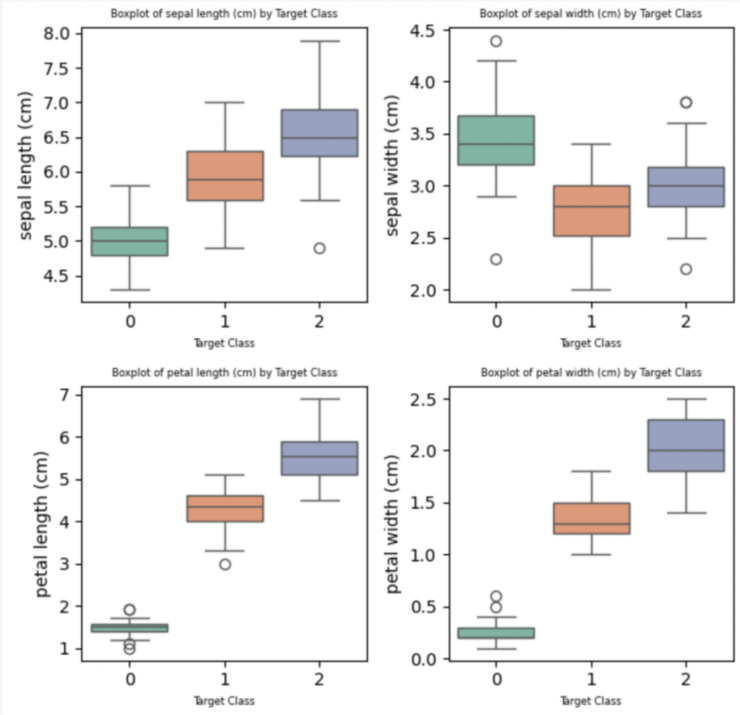
03

변수 선택

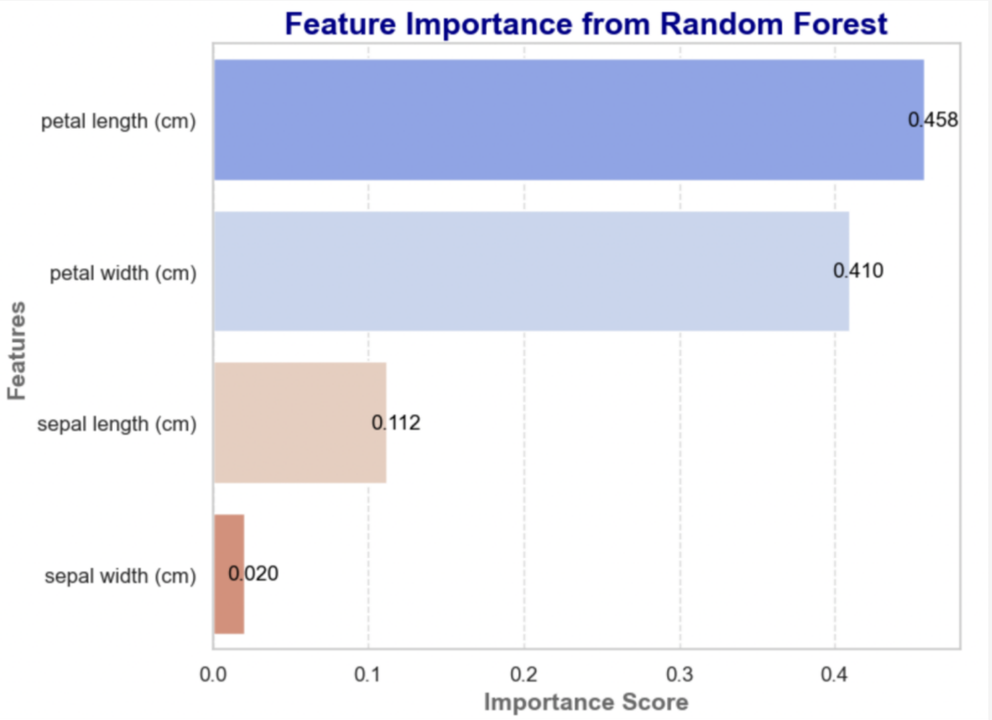
시각화 이용



상관계수



ANOVA



Feature Importance

03

차원 축소

차원 축소 방법

1. PCA 고차원 데이터를 저차원으로 변환하며 데이터의 분산을 최대화

```
from sklearn.decomposition import PCA

pca = PCA(n_components=2).fit_transform(X) # X는 입력 데이터, 2차원으로 축소
```

2. t-SNE 비선형 차원 축소 - 데이터 간의 유사성을 보존하며 저차원으로 변환

```
from sklearn.manifold import TSNE

tsne = TSNE(n_components=2).fit_transform(X) # X는 입력 데이터, 2차원으로 축소
```

03

차원 축소

차원 축소 방법

3. LDA 선형 판별 분석 - 주로 분류 문제에서 사용

```
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis

lda = LinearDiscriminantAnalysis(n_components=2).fit_transform(X, y) # X는 입력 데이터, y는 클래스 레이블
```

4. Autoencoder 신경망 기반 차원 축소 기법

```
from keras.models import Model
from keras.layers import Input, Dense

input_data = Input(shape=(X.shape[1],)) # 입력 데이터의 차원
encoded = Dense(2, activation='relu')(input_data) # 2차원으로 인코딩
autoencoder = Model(input_data, encoded)
autoencoder.compile(optimizer='adam', loss='mse')
autoencoder.fit(X, X, epochs=50, batch_size=16) # 입력 데이터를 재구성하는 방식으로 훈련
```

03

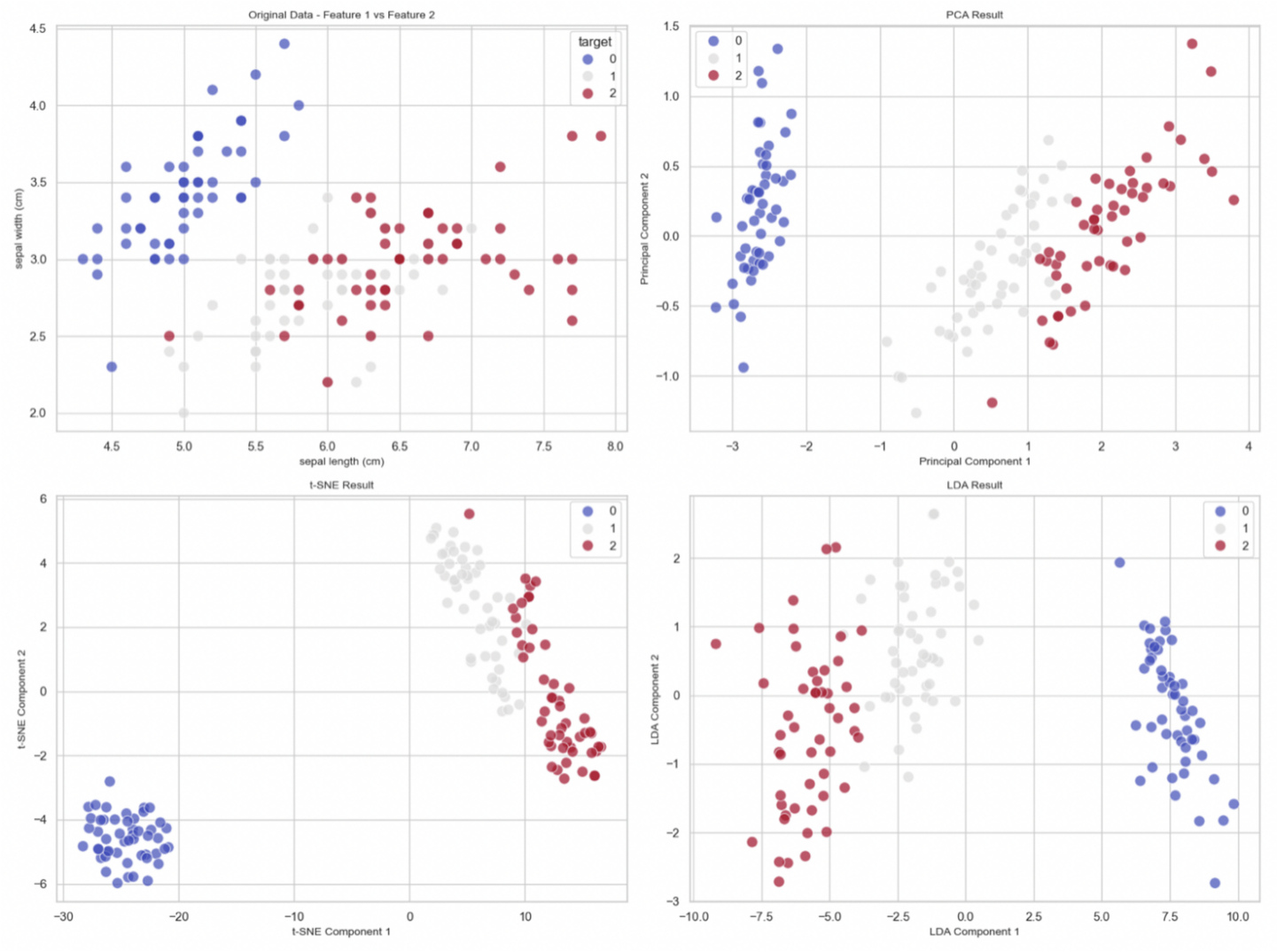
차원 축소

원본

PCA

t-SNE

LDA



CONTENTS

04

파생변수

기존의 변수로부터 생성된 **새로운** 변수

데이터에 추가적인 정보를 제공하고,

모델의 학습 효과를 향상하는 데 도움

04

파생변수

파생변수 생성

1. 수치적 변형

2. 범주형 변수 변환

3. 날짜/시간 변형

4. 텍스트 데이터 변형

04

파생변수

예시

1. 금융 데이터

자산 = 현금 + 주식 + 채권
부채 비율 = 총 부채 / 총 자산

2. 마케팅 데이터

고객 충성도 점수 = 구매 빈도 * 평균 구매 금액
구매 전환율 = 방문자 수 / 구매자 수

3. 의료 데이터

BMI = 체중 / (신장^2)
혈압 변동률 = (현재 - 이전) / 이전 혈압

04

파생변수

고려할 점

1. 과적합 너무 많은 파생변수를 생성할 경우, 모델 훈련 과정에서 과적합 가능

2. 의미 있는 데이터 생성한 변수가 의미 있어야 하며, 해석 가능해야 함

3. 변수의 상관관계 기존 변수와 강한 상관관계를 가진다면, 변수 선택 필요

04

파생변수

결국 가장 중요한 것은

데이터 도메인 지식을 잘 이해하고,

의미 있는 파생변수를 생성해야 함

CONTENTS

05

스케일링

변수의 **범위를 조정**하여

모델의 학습 효율성을 높이고 성능을 향상시킴

05

스케일링

필요성

1. 모델 수렴 속도 향상

2. 거리 기반 모델의 성능 향상

3. 데이터의 해석 용이성

05

스케일링

스케일링 기법

1. 정규화 각 피쳐의 값을 0과 1사이로 조정

2. 표준화 각 피쳐의 평균을 0, 표준편차를 1로 조정

3. 로버스트 스케일링 중앙값과 사분위수를 사용하여 스케일링

05

스케일링

스케일링 기법

1. 정규화

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}}$$

```
from sklearn.preprocessing import MinMaxScaler, StandardScaler, RobustScaler

min_max_scaler = MinMaxScaler()
data_normalized = min_max_scaler.fit_transform(data)
```

2. 표준화

$$X' = \frac{X - \mu}{\sigma}$$

```
standard_scaler = StandardScaler()
data_standardized = standard_scaler.fit_transform(data)
```

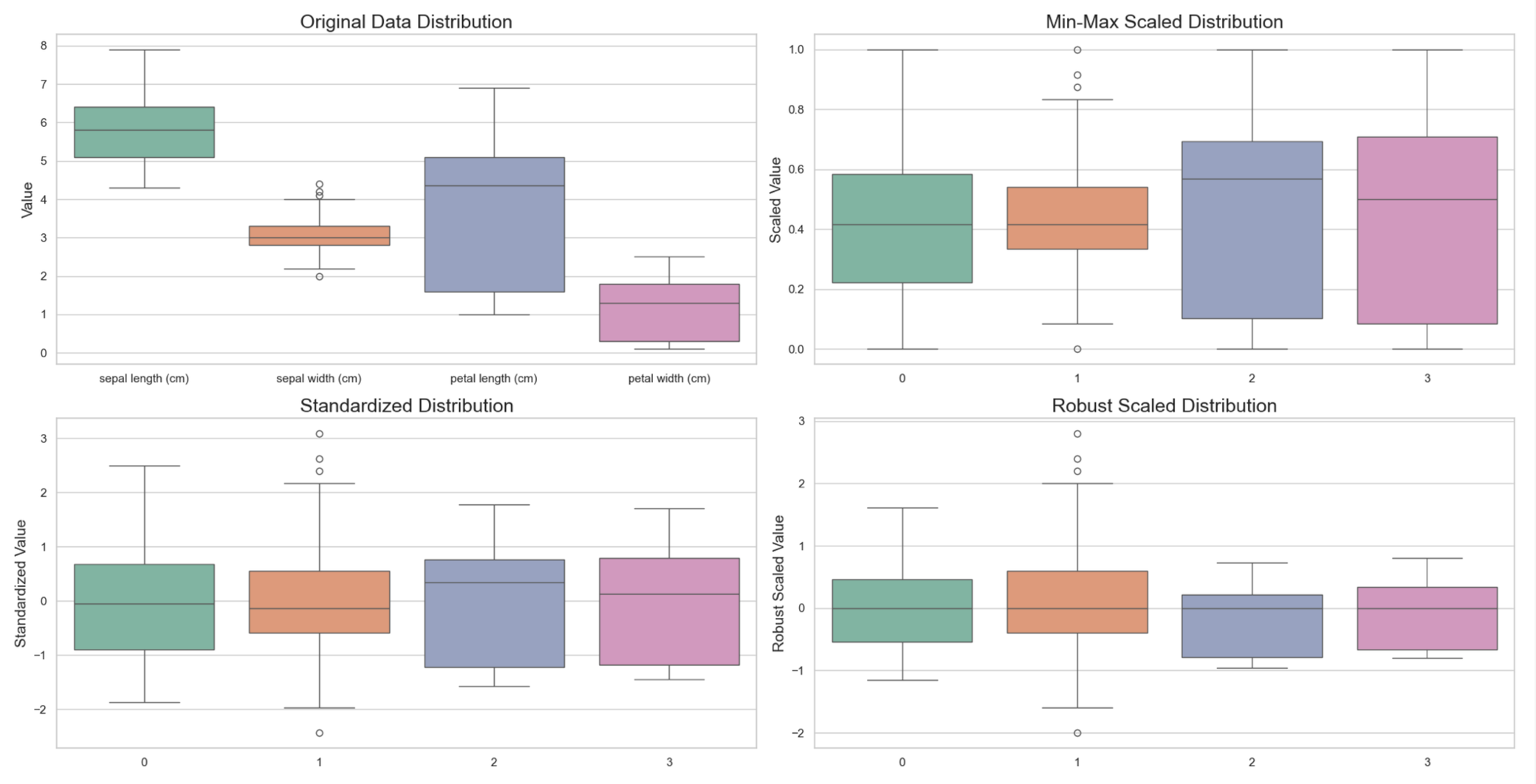
3. 로버스트 스케일링

$$X' = \frac{X - \text{median}}{IQR}$$

```
robust_scaler = RobustScaler()
data_robust_scaled = robust_scaler.fit_transform(data)
```

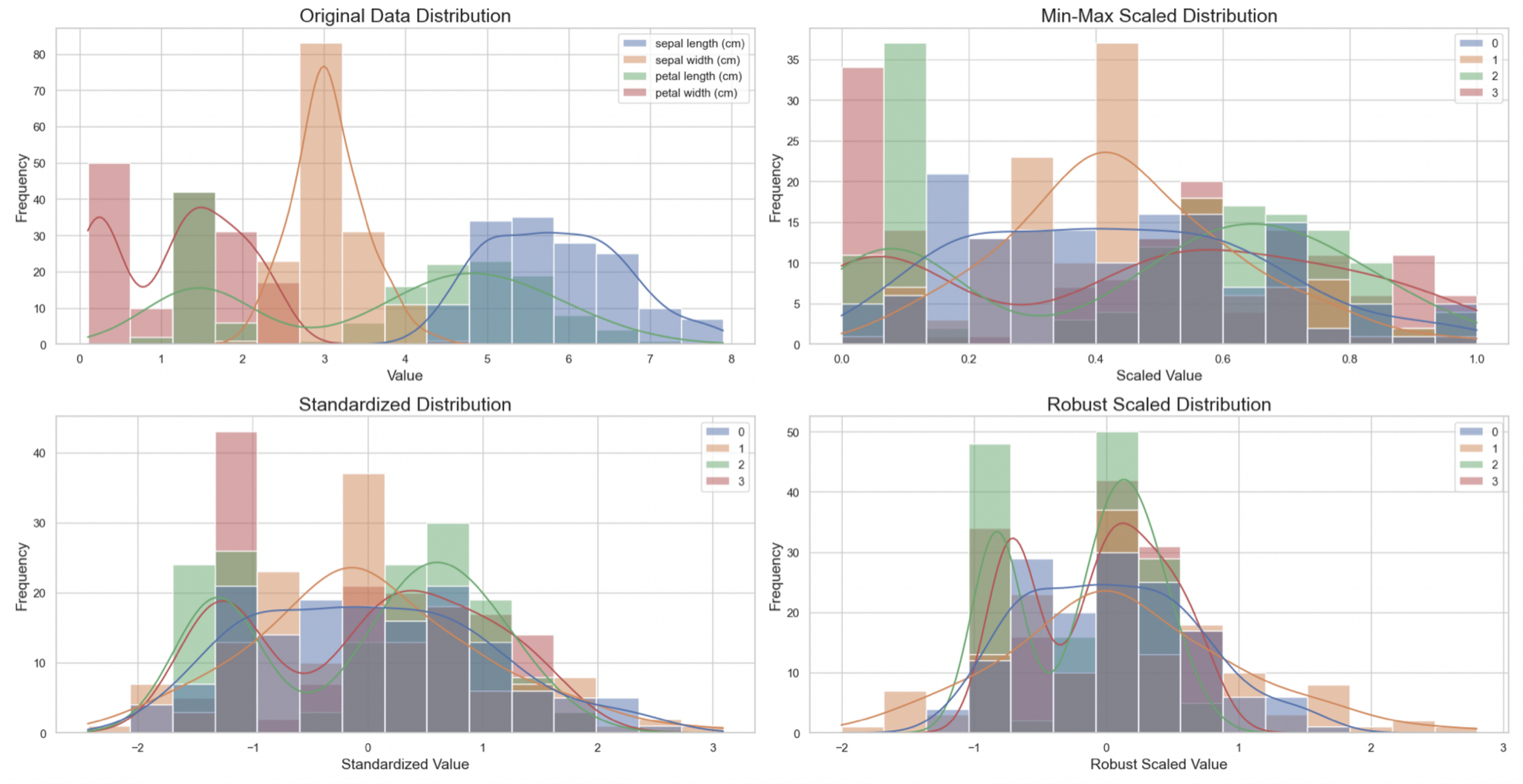
05

스케일링


Original Data Distribution
Value
sepal length (cm)
sepal width (cm)
petal length (cm)
petal width (cm)
Min-Max Scaled Distribution
Scaled Value
Standardized Distribution
Standardized Value
Robust Scaled Distribution
Robust Scaled Value

05

스케일링


Original Data Distribution
sepal length (cm)
sepal width (cm)
petal length (cm)
petal width (cm)
Frequency
Value
Min-Max Scaled Distribution
0
1
2
3
Frequency
Scaled Value
Standardized Distribution
0
1
2
3
Frequency
Standardized Value
Robust Scaled Distribution
0
1
2
3
Frequency
Robust Scaled Value

CONTENTS

06

결론

여러 기법을 통해 피쳐를 생성 & 변환하고,

도메인에 대한 이해를 통해 데이터 품질을 향상시킨다면,

높은 모델 성능을 기대할 수 있음