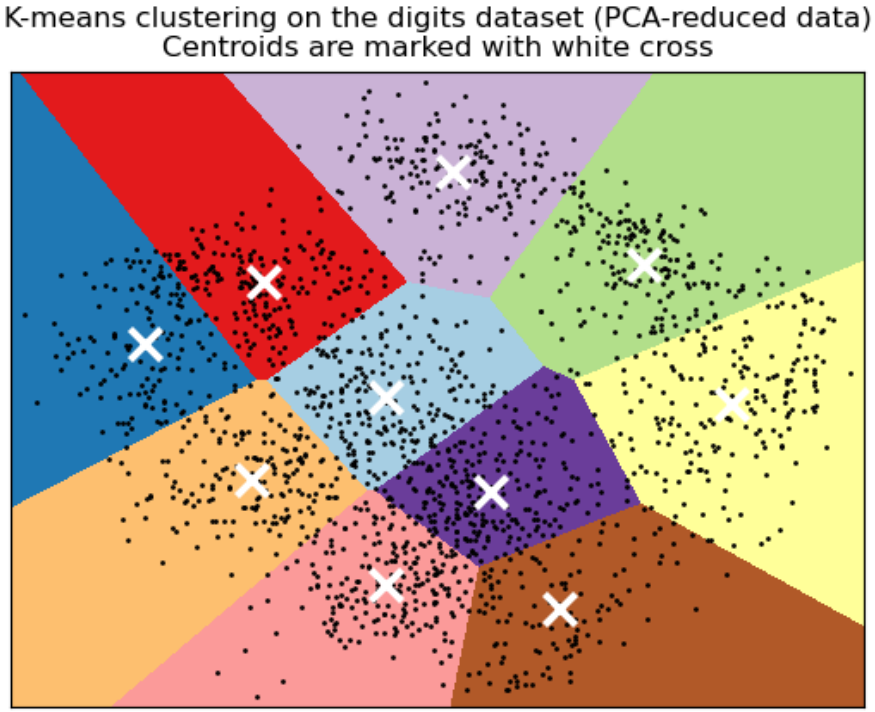
제10장. 군집분석(群集, Clustering)

- K-means Clustering -



Study Point

- 데이터, 알고리즘, 모형을 이해하고 scikit learn과 Keras API로 머신러닝을 이해한다.
- 회귀생성, 분류, 군집분석, 연관분석, 이상치 탐지, 시계열예측을 이해하고 실습한다.
- 회귀생성과 분류모형의 다양한 성능지표를 이해하고 실습한다.
- 교차검증과 하이퍼파라미터 튜닝 최적화를 이해하고 실습한다.
- 간단한 코드로 머신러닝을 적용하는 Low code 패키지인 PyCaret을 이해하고 실습

머신러닝 메커니즘(작동방식)

```
X = [[1,2,3], [11,12,13]]; y=[0,1]
```

```
from sklearn.cluster import KMeans
clf = KMeans(n_clusters=2)
clf.fit(X)
cls.transform(X)
clf.predict([[4,5,6]])
```

1. 데이터 준비
(Arrange Data)

2. 알고리즘 선택
(Choose an Algorithm)

k _ features

n _ instances

$$\begin{bmatrix} X_{11} & X_{12} & \dots & X_{1k} \\ X_{21} & X_{22} & \dots & X_{2k} \\ \vdots & \vdots & \vdots & \vdots \\ X_{m,1} & X_{m2} & \dots & X_{mk} \end{bmatrix}$$

❶ Data X, y

❷ Import Estimator
(Scikit-learn or Keras Estimator API)

3. 학습
(Learn)

❸ algorithm = Estimator()
model = algorithm.fit(X)

4. 예측
(Predict)

❹ model.transform(X)
model.predict(X)

Clustering

```
# Target의 label 을 안다고 가정하여 예측
from sklearn.datasets import load_breast_cancer
breast = load_breast_cancer()
breast.keys()

# 가상 데이터
import pandas as pd
data = pd.DataFrame(breast[ ' data ' ], columns=breast[ ' feature_names ' ])
data[ ' class ' ] = breast[ ' target ' ]
data.head()

# 훈련과 시험데이터로 구분하여 시험데이터의 지도학습 정확도
X = data.drop(columns=['class'])
y = data['class']
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=17)

from sklearn.ensemble import RandomForestClassifier
rf = RandomForestClassifier()
rf.fit(X_train, y_train).score(X_test, y_test)
```

```
0.965034965034965
```

Clustering

```
# Target의 label 을 모르고(즉 label이 몇 개인지 모른다) 훈련데이터로 학습
from sklearn.cluster import KMeans
kmeans = KMeans(n_clusters=2, random_state=17)              # 군집이 2개 일거라고 가정
X_clusters = kmeans.fit_predict(X_train).reshape(-1, 1)
cluster = pd.DataFrame(X_clusters, columns=['분류군집'])
cluster.head()

cluster['실제군집'] = y_train
cluster['정답'] = cluster['실제군집'] - cluster['분류군집']
cluster.head()
cluster['정답'].value_counts(normalize=True)
```

```
 0.0    0.572755
-1.0    0.318885
 1.0    0.108359
Name: 정답, dtype: float64
```

```
# 시험데이터로 군집분석 후 정답 확인(정확도)
kmeans.predict(X_test)
y_test
accuracy = kmeans.predict(X_test) == y_test
accuracy
print(f'군집분석의 정확도 : {accuracy.sum()/len(y_test)}')
```

```
군집분석의 정확도 : 0.8951048951048951
```