https://doi.org/10.7840/kics.2024.49.9.1306

데이터베이스의 마이크로서비스 전환을 위한 K-Means 군집화 활용 분류·분석 자동화 시스템 설계 및 구현

박 선 철◆, 김 영 한°

A Design and Implementation of Automated Classification and Analysis System Using K-Means Clustering for Transition of Database to Micro-Services-Architecture

Sun-chul Park◆, Young-han Kim°

요 약

통합 데이터베이스(DB: Database)의 클라우드 전환은 복잡한 구조의 통합DB를 서비스 용도에 따라 테이블 단위로 분류하는 것을 기본으로 한다. 분류과정은 대부분 숙련된 전문가의 수작업을 통해 진행되는데, 본 논문에서는 머신러닝의 군집화 알고리즘을 적용한 자동화 시스템을 설계하고 검증하였다. 제안하는 시스템은 DB의 SQL(Structured Query Language) 발생빈도를 이용한 통계적 처리를 통해 시계열 데이터를 생성하고 머신러닝에 적합한 형태로 변환 후 K-Means 군집화 알고리즘을 적용하여 자동화하였다. 제안된 시스템의 성능 검증은 기존의 임의 초기분류방식과 자동화 분류의 결과 비교와 K-Means의 최적 파라미터 확인을 위한 엘보우 알고리즘 및 군집화 성능 결과 분석등을 수행하였고 이를 통해 제안된 방식의 자동화 효과를 확인하였다.

키워드 : K-Means, 비지도학습, 머신러닝, 클라우드 마이그레이션, MSA, DB분석
Key Words : K-means Clustering, Unsupervised Learning, Cloud Migration, MSA, DBMS Analysis

ABSTRACT

The most challenging task in transitioning an integrated database to the cloud is categorizing tables according to the structure. We automated this using machine learning clustering, collecting SQL frequencies, generating time-series data, and applying K-Means. We compared the proposed automation method with manual classification, identified K-Means parameters using Elbow, and showed the automation's effectiveness.

Ⅰ. 서 론

DB의 클라우드 전환을 위한 최적화 방식은 표 1에 분류한 것처럼 단순이전보다 DB 재구조화 등이 요구되는 복잡하고 고난이도 작업이다. 통합DB는 테이블 구조가 수십개에서 수백개로 구성되며, 자료의 볼륨도 최소 기가바이트에서 테라바이트에 이르는 대형 DB이다. 이러한 통합DB의 재구조화는 DB에서 시작해 비지니스 로직과 연관된 서비스 분석을 통해 변경 후에도 데이터 품질이 유지되도록 분류하기위한 분석 과정이 매우 중요하다. 통합DB는 단일 DB에 비해 많은 비지니스 서비스와 연결되어 있어 분석도 수일에서 수주가 소요

◆ First Author : Soongsil University Graduate School of AI IT Convergence, sunpark@soongsil.ac.kr, 학생회원
° Corresponding Author : Soongsil University School of Electronic Engineering, younghak@ssu.ac.kr, 종신회원
논문번호 : 202404-064-C-RN, Received April 11, 2024; Revised June 4, 2024; Accepted June 11, 2024

표 1. 클라우드 마이그레이션 성숙도 레벨
Table 1. Level for maturity of Cloud Migration

Level	Type	Description
Low	Lift and Shift	Transitioning the existing environment to virtual(P2V) Re-Hosting
High	Optimizing	Reconstructing through re-installing, restructuring, and redevelopment Re-Engineering

되는 중대형 프로젝트이다. 이러한 특성으로 DB 분석 분야는 문서 중심의 수작업 분석으로 수행되나, 오랜기간 잦은 변경에 대한 문서화 부재의 문제와 대상 서비스의 각기 다른 특성으로 인한 범용화의 어려움 등으로 전환 관련 자동화 기술은 데이터 원격복제 수준에 머물러 있다.

PaaS(Platform as a Service)에서의 마이크로서비스아키텍처(MSA)는 그림 1과 같이 협력DB (Aggregation DB) 모델로 배치한다. A. Mendelsohn은 기업 RDB 시장에서 오라클 DBMS의 대용량 집계처리와 분산처리 등 고성능 기능으로 점유율 1위를 유지하고 있다고 하였다[1]. B. J. Kim은 클라우드 서비스 사업자(CSP)가 제공하는 DBMS 솔루션 현황을 분석한 결과 오라클 DB 같은 상용보다 MariaDB 같은 오픈소스 DB가 널리 사용되고 있다고 분석하였다[2]. 따라서, 레거시 통합DB의 클라우드 MSA 전환은 대용량의 통합DB를 다수의 소형DB로 재구조화하는 것이 필수적이다. 이를 위해 통합DB의 데이터 품질을 보장하면서 필수적인 DB만 클라우드로 전환하여 총 운영비용이 절감되도록 하는 것도 중요한 고려사항이다.

본 연구는 통합DB의 클라우드 MSA로 전환시 수작업 중심의 DB 분석과정 중 분류를 자동화하는 시스템 구현을 목표로 한다. 일반적인 DB 마이그레이션은 업그레이드, 서비스 고도화, 이기종 DB전환, 통합DB 분리 등에 그림 2와 같은 절차로 대상의 특성에 맞춘 분석을 수행한다. 더욱이 클라우드로의 DB 전환은 관련한

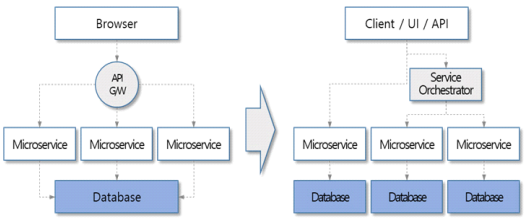

그림 1. PaaS에서의 통합DB 모델과 협력DB 모델 아키텍트
Fig. 1. Integrated DB and Aggregation DB Model in PaaS

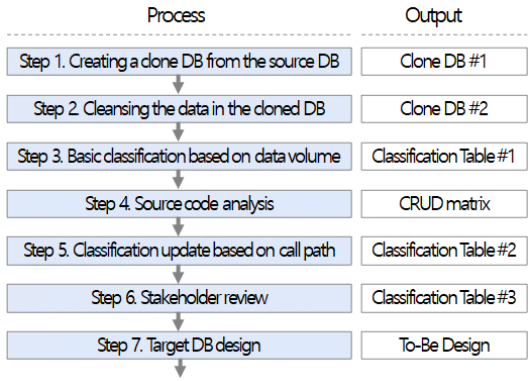

그림 2. 일반적인 DB 마이그레이션 분석 절차
Fig. 2. Typical Database Migration Analysis Procedure

연구도 많지 않고 있더라도 프레임워크 등 절차연구에 집중하고 있다. 이에 본 연구에서는 통합DB의 이력 데이터를 수집 후 가공하여 유형별로 자동 분류하는 시스템을 설계하고 검증한다. 본 연구의 결과로 수작업 중심의 DB 분석 분야에도 시스템화 및 자동화의 가능성을 증명하고 수행절차 및 시간을 단축하는 신속성과 효율성을 향상시켜 레거시 통합DB의 클라우드 마이그레이션에 효과적일 것이다.

Ⅱ. 관련 연구

2.1 클라우드 DB 마이그레이션

통합DB 전환은 데이터 볼륨도 크고 복잡성도 높아 데이터 품질을 유지하는 것이 매우 중요하다. A. A. Hussein은 DB 전환 프로젝트 성과분석에서 약 84%가 기간초과, 예산초과로 실패한 프로젝트였는데, 그 원인을 변경 이력관리 부실, 분석 부족, 데이터 정제문제 등 DB 분석 단계의 미흡함을 큰 요인으로 지적하였다. 이의 방지를 위해 소규모 마이그레이션을 반복하는 애자일 원칙을 권고하였다[3]. 하지만, DB의 정합성을 간과한 애자일 방식은 데이터 품질을 저하시킬 위험이 있다. M. Barbosa et al.은 다수의 MSA DB전환 연구에서 기존 연구는 전환 프로세스와 전환 프레임워크를 대상으로 하고 DB 아티팩트를 다루는 연구가 존재하지 않는데, 비즈니스 로직을 이해해야 하는 분석의 한계를 원인으로 보았다[4]. 비즈니스 로직은 분야별 전문가의 영역이고 서비스마다 고유한 특성을 이해해야 하는 어려움 때문이라는 문제점을 지적하였으나 이의 해결방안을 제시하지 못한점은 아쉽다.

레거시 통합 DB의 대용량 저장소의 문제도 있다. M. Stonebraker et al.은 대용량 통합 DB를 복제하여

이전하면 스토리지 공간 비용과 복제본 문제가 발생하는데 클라우드에서 대용량을 유지하기 위해 보다 저렴한 빅데이터 저장소인 HDFS(Hadoop Distributed File System)와 같은 분산 파일시스템이나 인메모리 DB를 사용하는 방식을 제안하였다[5]. 하지만, 이런 방식은 RDB 고유 파일처리 알고리즘 변경 없이는 적용할 수 없다. M. Ellison et al.은 기존 클라우드 전환 연구가 소프트웨어 구성 측면에 맞춰져 있어 DB 전환에 대한 비용 및 기간 예측이 어렵다고 지적하고, DB 로그 및 스키마에서 비용 및 기간 추정치를 얻는 연구를 수행하였다[6]. 하지만, 예측이 단순히 원격복제를 위한 전송시간과 요금 계산에 중점을 두고 있다.

Z. Wang et al.은 레거시 환경의 아키텍처 설계 결함을 가장 큰 장애요소로 지적하고, 전환이 성공하기 위해서는 전환 방법과 적용 패턴모델의 중요성을 강조하였다[7]. 클라우드 DB 배치모델을 선택할 때 통합DB의 단순 이주방식도 고려될 수 있다. 하지만, V. Velepucha et al.은 모놀로식 아키텍처를 MSA로 전환할 때 통합 DB 모델을 사용하면 안티패턴이 될 수 있어 한시적 제한적으로 사용토록 주의가 필요하다 하였다[8]. 연구에서는 설계방식을 다루고 구체적으로 분산하는 방법을 다루지 않는 것은 아쉽다. 따라서, MSA의 DB 전환은 대용량, 고비용의 상용 DBMS에서 다수의 오픈소스 DB로 분산 소형화하여 안티패턴을 방지하고 전환 총비용을 최적화하기 위해 데이터를 분리하는 것이 매우 중요하다.

2.2 비지도 학습 및 K-Means 군집화

비지도 학습은 데이터의 특성을 알기 어려울 때 특성분석에 주로 사용되어 군집분석, 시각화, 차원축소, 연관 규칙 학습 등에 활용된다. 비지도 학습 알고리즘 중 K-Means는 데이터 식별 알고리즘으로 영상분할, 구매성향, 문서 유사성 분류 등과 물류분야의 상권, 거점지 선정 등에도 널리 활용되고 있다. K-Means는 그림 3처럼 중앙값과 유클리드 거리 계산 반복을 통해 유사성 높은 데이터끼리 분류하는 군집화 알고리즘이다.

표 2는 K-Means 수행 절차로 1, 2단계를 설정하면, 모든 데이터 군집 할당이 완료될 때까지 3, 4단계를 스

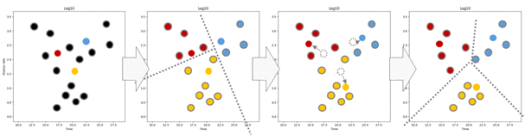

그림 3. K-Means 군집화 알고리즘 동작 방식
Fig. 3. Mechanism of the K-Means Clustering

표 2. K-Means 군집화 단계별 적용 알고리즘
Table 2. K-Means Step-by-Step Application Algorithm

Step	Procedure	Algorithm
1	Setting the number of clusters	- Rule of Thumb - Elbow Method - Info-Criterion Approach
2	Selecting initial centroids	- Randomly Select - Manually Assign - K-means++
3	Assigning data to clusters	- Euclidean distance
4	Calculating after moving centroids	-

스로 반복하여 수행한다.

K-Means는 최적의 K 선정 문제와 대량 데이터 성능 문제, 결과에 대한 성능평가의 어려움의 3가지 단점이 있다. A. M. Ikotun et al.은 잘못된 K로 인한 지역수렴 문제를 해결하기 위한 Fuzzy, Rough, Overlapping Cluster 등의 다양한 개선 알고리즘을 제시하였다[9]. R. Mussabayev et al.은 대량 데이터 성능 문제를 해결하기 위해 데이터를 분해 후 결합 방식으로 접근하는 유클리드 최소제곱 클러스터링 알고리즘을 제안하였다[10]. 그 외 널리 사용되는 군집화 성능평가는 내부평가와 외부평가로 구분되며 표 3과 같은 알고리즘이 사용된다.

표 3. K-Means 군집화 평가 척도
Table 3. K-Means Clustering Evaluation Algorithm

Measurement	Algorithm
Internal measurement	- Davies-Bouldin index - Dunn index - Silhouette coefficient score
External measurement	- Rand measure - F measure - Jaccard index

2.3 엘보우 알고리즘(Elbow method)

K-Means의 최적의 K를 찾기 위한 다양한 접근방법이 있다. T. m. kodinariya와 R. Makwana는 널리 사용되는 관련 6가지의 알고리즘을 비교분석 하였다[11]. 그 중 엘보우 알고리즘은 오래되고 널리 사용되는 시각화 알고리즘으로 특정 값에서 급격히 변화한 후 정체되기 시작하는 지점을 최적의 K로 판정한다.

2.4 실루엣 계수(Silhouette Coefficient)

실루엣 계수는 군집화 결과 검증 알고리즘이다. 모든 대상 요소 간 거리행렬로 개별 실루엣 계수를 구한다.

실루엣 계수는 0에 가까울수록 군집화가 잘 되었음을 의미한다. 실루엣 스코어는 개별 실루엣 계수의 평균으로 1에 가까울수록 좋은 군집화다. P. J. Rousseeuw는 실루엣 계수를 시각화하는 방식을 제안하였는데 결과를 블록으로 표현하면 각 군집 간의 분포 비교를 용이하게 할 수 있다고 하였다[12]. 이번 연구도 결과 비교에 블록화 표현을 사용한다.

Ⅲ. 제안 기법

3.1 제안 시스템

본 연구는 통합 DB에 저장된 여러 테이블을 용도별로 식별하는 자동화 시스템 구현을 목적으로 한다. 그림 4처럼 각 테이블의 빈도를 수집 후 전처리한 후 유형별 군집 분류 자동화를 통해 용도를 식별한다.

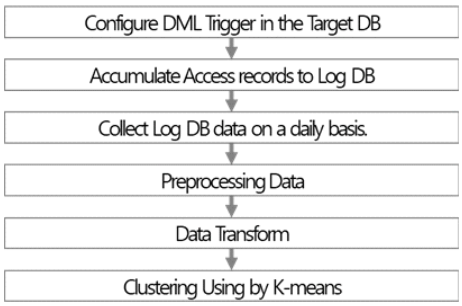

그림 4. 시스템 처리 흐름도
Fig. 4. System flowchart

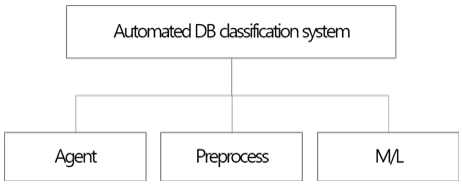

그림 5. DB분류 자동화시스템 구성도
Fig. 5. DB Classification Automated System Architecture

표 4. 시스템 부문별 역할
Table 4. Roles of System Sub module

Sub system	Role
Agent	Connect to the DB, create a history database, and set up a DML tracking environment for the table.
Preprocess	Process the data stored in the history database into a format suitable for machine learning
M/L	Apply machine learning algorithms to cluster the Table.

단계별 처리에 맞춰 DB분류 자동화 시스템을 설계하고, 그림 5처럼 3가지 역할로 나누었다. 표 4는 하위 시스템의 역할이다.

3.2 DML 트래킹(Tracking)

DB의 빈도 수집을 위해 SQL 명령어를 사용한다. SQL 명령어는 DDL(Data Define Language), DML(Data Manipulation Language), DCL(Data control Language), TCL(Transaction Control Language)로 나뉘며 본 연구에서는 DML을 이용한다. DML 트래킹은 DBMS의 트리거를 이용하는데, DB의 DML 이벤트에 자동 반응하는 기능으로 그림 6과 같은 문법으로 활성화 할 수 있다.

이력용 테이블 스키마는 표 5와 같이 정의하였다. 대상 테이블 정보와 테이블의 정보접근이 발생한 시간을 기록한다. 시간 정보는 동일한 테이블의 이벤트를 24시간 분포로 변환하기 위해 사용한다.

```
{CREATE | RECREATE | CREATE OR ALTER} TRIGGER name FOR {table_name | view_name}
 [ACTIVE | INACTIVE]
 {BEFORE | AFTER}
 {[INSERT [OR UPDATE] [OR DELETE] | UPDATE [OR INSERT] [OR DELETE] | DELETE [OR UPDATE] [OR INSERT] }
 [POSITION n] AS
BEGIN
  ...
END
```

그림 6. 트리거 스크립트 문법(출처: WIKIPEDIA)
Fig. 6. Trigger Syntax (Source: WIKIPEDIA)

표 5. DB 이력저장용 TB_DATA_RAW 스키마
Table 5. Schema for TB_DATA_RAW

Filed Name	Attribute	Description
SEQ	Number	index, Primary-Key
TIME_STAMP	Date	event timestamp
DB_NAME	Varchar(32)	target DB name
TB_NAME	Varchar(64)	target Table name

```
1. 테이블 로그데이터 수집용 테이블
-- DROP TABLE CLOUD.TB_DATA_RAW ;
CREATE TABLE CLOUD.TB_DATA_RAW (
    SEQ number not null, TIME_STAMP date, DB_NAME varchar2(32),TB_NAME varchar2(64)
) ;

2. 시퀀스 생성
-- DROP SEQUENCE CLOUD.SEQ ;
CREATE SEQUENCE CLOUD.SEQ ;

3. CRUD 별 트리거 생성
-- DROP TRIGGER CLOUD.TRI_ARCHIVE_TB_ITEM_I ;
CREATE OR REPLACE TRIGGER CLOUD.TRI_ARCHIVE_TB_ITEM_I
AFTER INSERT, UPDATE, DELETE
ON CLOUD.TB_ITEM
FOR EACH ROW

BEGIN

INSERT INTO CLOUD.TB_DATA_RAW2
VALUES (SEQ.NEXTVAL, SYSDATE, 'ARCHIVE', 'TB_ITEM') ;

END TRI_ARCHIVE_TB_ITEM_I ;
```

그림 7. 이력 저장 테이블 및 트리거 정의 SQL문
Fig. 7. SQL for Tracking Table and Trigger

다음은 이력 DB와 대상 테이블에 대한 DML 트리거를 설치한다. 그림 7은 DB에 대한 INSERT, UPDATE, DELETE 이벤트가 발생하면 이력DB에 기록하도록 구성하였다.

3.3 시계열 데이터의 좌표계 변환

이력 DB를 통해 수집한 데이터는 시계열 데이터로 군집화에 사용하는 좌표계 데이터와는 형태가 다르다. 군집화 알고리즘에 적용하기 위해서는 시계열 특성을 대표하는 단일 좌표계 데이터로 변환해야 한다. 연속된 수치에서 대표성을 갖는 단일값을 얻는 방법으로는 평균, 중앙, 최대, 최빈 취득 등이 사용된다. 본 연구는 그림 8처럼 다양한 시계열 형태를 고려해 무게중심을 데이터변환에 적용했다.

삼각형 무게중심 계산에는 3개의 꼭짓점 좌표가 필요하다. 본 연구에서는 이력DB 데이터의 발생시간을 x축으로 y축은 접근로그 데이터의 평균취득을 적용했다. 무게중심 좌표를 구하는 점 A, B, C를 수식(1)과 같이 정의한다.

$$\begin{aligned} V_n &= number\,of\,data\,occurrences\,in\,n\,hours \\ T_{\max} &= time\,when\,V\,occurred\,most\,frequently \\ A=(T_s, V_s), B&=(T_{\max}, V_t), C=(T_e, V_e) \end{aligned} \quad (1)$$

무게중심 점 G는 수식(2)처럼 계산한다.

$$\begin{aligned} G &= \left(\frac{|x_1|+|x_2|+|x_3|}{3}, \frac{|y_1|+|y_2|+|y_3|}{3}\right) \\ &= \left(\frac{k}{3}, \frac{p}{3}\right) = (Avg(k), Avg(p)) \end{aligned} \quad (2)$$

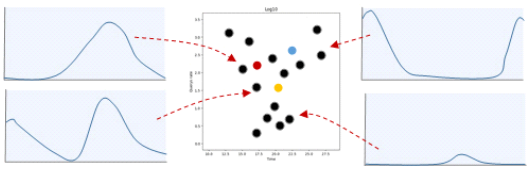

그림 8. 시계열 데이터의 좌표계 변환 문제
Fig. 8. Example for Data-Transform of time series

Ⅳ. 구현 및 시험평가

4.1 시험 개요

본 연구의 성능시험은 기업 통합DB에 저장된 데이터로 검증을 수행했다. 시험을 위한 데이터는 통합DB에 테이블 접근 기록 데이터를 사용하여 각 DB별 사용빈도로 변환하였다. 변환된 빈도데이터를 전처리를 통해 K-Means를 적용할 수 있는 데이터 형식으로 변환 후 사람에 의한 임의분류와 연구의 군집분석 결과의 일치성을 비교하고, 실루엣 계수로 품질을 측정하여 효과성을 검증한다.

4.2 시험환경 구성

시험환경은 표 6과 같다. DML 트래킹은 서버환경에서 군집분석은 구글 코랩에서 수행하였다.

표 6. 시험환경
Table 6. Test environment

Type	Description
CPU/RAM	Intel 2.4Ghz 4Core / 8GB
O/S	Microsoft Windows Server 2016 Std.
DBMS	Oracle 18c Express Edition
K-Means	Google colab and sklearn.cluster lib.

4.3 시험 데이터

시험 데이터는 기업내 통합DB에 저장된 이력 데이터를 이용하여 추출하였으며, 표 7과 같이 분석에 사용할 TB_DATA_RAW 테이블에 저장된 이력 데이터 약 4천만건을 대상으로 했다.

시험 데이터는 기업 서비스, 사용 분석, 계정등 관리 데이터의 총 3가지 유형으로 표 8과 같은 유형 특성이 있다. 이번 연구에서는 시험 데이터 특성에 맞추어 3가지 유형으로 분류하는 것을 목적으로 한다.

표 7. 시험 데이터 요약
Table 7. Summary of test data

Type	Description
Range	2001/01/01~2023/12/08
Records	39,855,537

표 8. 시험데이터 유형에 따른 빈도 발생시기
Table 8. Frequency of occurrence based on data type

Type	Purpose	Occurrence Time(e.g.)
Service Data	The data for the service shows a high frequency during working hours.	09~20 hr.
History Data	A scheduler that starts after work and ends in the early morning for service analysis.	19~06 hr.
Account Data	There is no specific cycle, but it mainly occurs during working hours.	non-structured

4.4 데이터 전처리

전처리는 총 2단계로 1단계는 DB 단위 평균 빈도 계산이다. DB_NAME과 TB_NAME을 복합키로 하여 데이터 시간별로 발생빈도를 1시간 간격 빈도평균으로 계산한다. 이 과정을 통해 수천만건의 이력 로그가 243개의 목적 테이블에 대한 시간대별 활성빈도로 변환된다. 2단계는 데이터 회전이다. 표 8에서 보듯이 시험 데이터는 대부분의 데이터가 9시경에 데이터 호출이 시작되는 데이터임을 알수 있다. 이런 특성을 반영하여 배치처리를 묶음하기 위해 x값에 왼쪽회전을 적용하여 시작을 9시로 이동한 후 수식(1)에 정의된 꼭지점 A, B, C를 계산 하였다.

4.5 데이터 적재

데이터 적재 및 처리는 그림 9와 같이 판다스(Pandas) 데이터프레임을 이용하여 처리하였다. 데이터프레임에는 약 4천만건의 원본 데이터에서 일일 데이터로 변환된 243개 테이블에 대해 계산된 점 A, B, C가 저장되어 있다.

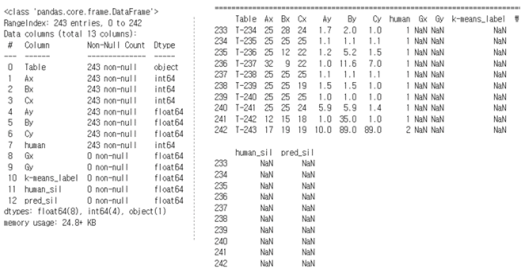

그림 9. Pandas로 로딩한 Dataframe
Fig. 9. Dataframe loaded by Pandas

4.6 무게중심 좌표 계산 및 로그 스케일 적용

무게중심 점 G(x,y)는 점 A, B, C의 무게중심 좌표 계산식에 따라 계산하였다. 그림 10은 계산된 G(x,y)에 로그 스케일을 적용 후 좌표계에 표시한 그래프다. 세로축은 다수가 약 200 미만에 몰려있고, 가로축은 18~20 사이에 발생이 줄어드는 것을 알 수 있다. 이를 통해 검증을 위한 기준으로 세로축 2.0과 가로축은 20이라는 데이터를 획득하였다.

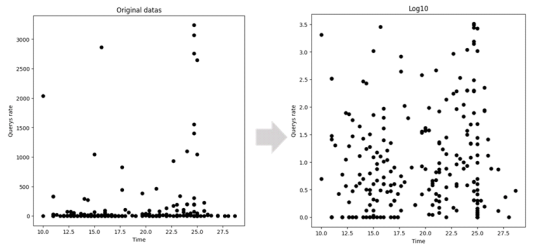

그림 10. 무게중심 좌표에 로그 스케일 적용 전후 비교
Fig. 10. Before and after applying log scale to centroid

4.7 검증 데이터 생성

비지도 학습 군집화 이후 결과를 검증하기 위해 성능검증용 분류를 생성한다. 앞선 로그스케일 과정에서 확인된 시험 데이터의 특성에 맞추어 그림 11처럼 임의분류 스크립트로 검증용 비교데이터를 생성하였다. 0은 관리 데이터, 1은 로그 데이터, 2는 서비스 데이터로 분류했다.

그림 12는 검증을 위한 임의분류 결과 그래프다. 검증용 임의분류가 계획과 같이 세로축 2.0과 가로축 20을 기준으로 데이터가 제대로 분리되었음을 확인할 수 있다.

```
def make_human_checker(Gx, Gy):
  if Gy < 2.0 :
    if Gx < 20 : # 20hour
      return 0
    else
      return 1
  return 2
```

그림 11. 성능검증 데이터를 위한 분류코드 적용 코드
Fig. 11. Source-code for compare data

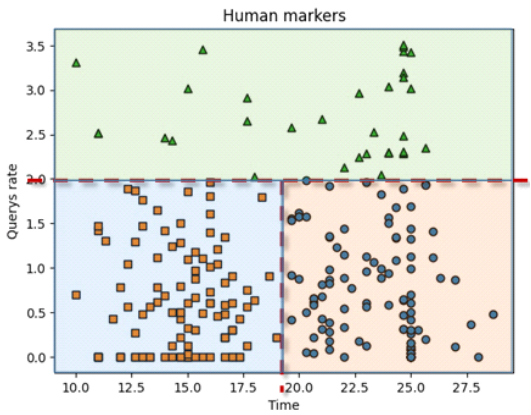

그림 12. 검증 데이터로 분류한 군집화 분포
Fig. 12. Cluster distribution classified with validation data

4.8 엘보우 포인트 계산

본 연구의 시스템은 알고리즘으로 최적의 값을 계산한다. 최적의 K를 위한 엘보우 포인트는 그림 13처럼 3에서 완만해지는 K값이 3임을 알 수 있다. 이를 통해 현재 적재된 데이터를 3가지 유형으로 분류할 수 있음을 알 수 있다.

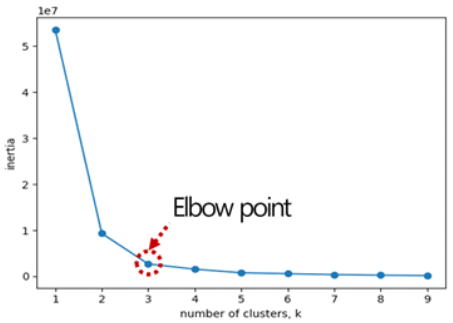

그림 13. 엘보우 포인트를 이용한 최적의 K 탐색
Fig. 13. Optimal K using the Elbow method

4.9 군집분석(Clustering)

K값 3과 점 G를 이용하여 표 9와 같은 파라미터로 K-Means 군집분석을 수행한다. 총 클러스터는 3으로 정의하였으며 최대 반복횟수는 군집이 완료되면 자동 종료되는 특성에 따라 큰값인 1,000으로 설정하였다.

구글 코랩에서 사이킷-런 라이브러리를 사용하여 그림 14처럼 K-Means를 수행한다.

그림 15는 K-Means 수행 결과가 저장된 데이터프레임이다. K-Means_label은 군집화 알고리즘을 통한 예측결과, human_sil에는 human 속성으로 계산된 실루엣 계수, pred_sil에는 군집화 결과로 계산된 실루엣 계수를 저장하였다.

군집화 결과의 성능 비교를 위해 임의분류와 예측분

표 9. K-Means 알고리즘 적용 파라미터
Table 9. Parameters applied in the K-Means algorithm

Parameter	Input Value
init	k-means++
n_clusters	3
max_iter	1000
random_state	None(=0)

```
import warnings
warnings.filterwarnings('ignore')
from sklearn.cluster import KMeans
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler

# 실루엣 분석 metric 값을 구하기 위한 API 추가
from sklearn.metrics import silhouette_samples, silhouette_score

X = np.stack([raw_dataframe['cx'], raw_dataframe['cy']], axis=1)

#kmeans = KMeans(init="k-means++", n_clusters=3, random_state=0)
center = [[0,0], [0, 100], [15, 2000]]
kmeans = KMeans(init="k-means++", n_clusters=4, max_iter=1000, random_state=0)

scaler = StandardScaler()
pipeline = make_pipeline(scaler, kmeans)
```

그림 14. 구글 코랩을 이용한 K-Means 소스코드
Fig. 14. Source code for K-Means using Google Colab

	Table	Ax	Bx	Cx	Ay	By	Cy	human	Gx	Gy	#
0	T-001	25	21	24	6.0	23.1	5.7	1	23.333333	1.064458	
1	T-002	32	32	19	3.0	3.0	1.0	1	27.666667	0.367977	
2	T-003	25	22	24	2.0	26.1	1.5	1	23.666667	0.994170	
3	T-004	25	25	24	2317.0	2317.0	32.8	2	24.666667	3.191898	
4	T-005	25	25	24	4118.0	4118.0	30.0	2	24.666667	3.440174	

	k-means_label	human_sil	pred_sil
0	2	0.494257	0.471570
1	2	0.368015	0.393977
2	2	0.510141	0.500488
3	1	-0.147186	-0.162818
4	1	-0.117190	-0.134490

그림 15. K-Means 군집화 수행결과
Fig. 15. Results of K-Means clustering

류 결과를 그래프로 비교하면 그림 16과 같다. 좌측 검증 데이터는 기계적인 분류로 세로축 2.0을 기준으로 상하가 명확히 구분되고 있으나, 우측 알고리즘에 의한 분류는 수작업에 의한 분류보다 경계의 특성을 더 세밀하게 분류하고 있음을 확인할 수 있다.

그림 17은 임의분류와 예측분류 간의 크로스탭 분석이다. 분류0에서 2개, 분류1에서 21개의 차이가 나타났으며, 표 10처럼 유사도 90.6%로 그간 사람에 의해 분류하던 임의분류를 대체하여 신속하게 처리할 수 있음

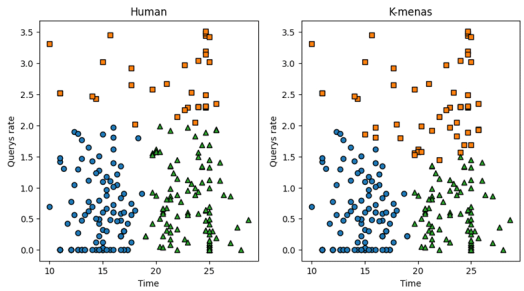

그림 16. 검증용 분류와 K-Means의 결과 비교
Fig. 16. Comparison between validation and K-Means

k-means_label	0	1	2
human			
0	99	4	0
1	0	34	0
2	2	17	87

그림 17. 검증분류와 K-Means 결과의 교차표 비교
Fig. 17. CrossTab between validation and prediction

표 10. 성능시험 수행결과
Table 10. Performance test results

Type	Measurement Value
All Record	243
Match Record	220
Mismatch	23
Matching rate	0.906

을 확인할 수 있다

4.10 실루엣 계수 측정

K-Means로 계산된 실루엣 계수와 실루엣 스코어를 그림 18처럼 그래프로 표시하니 K가 2와 3에서 적정한 분포로 도출되고 있으며, 실루엣 스코어는 2에서 0.661, 3에서는 0.548로 다소 차이가 있으나 본 연구에서 목표하는 3가지로 분류하는 것에 적용 가능함도 알 수 있다.

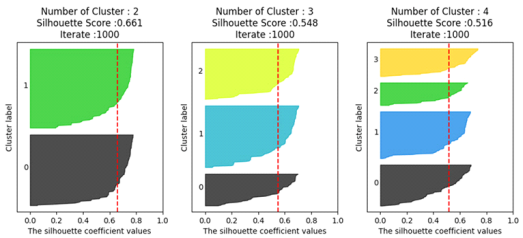

그림 18. 실루엣 스코어 블록 그래프
Fig. 18. Silhouette Score Block Graph

V. 결 론

본 연구는 단일 통합DB에 저장된 다양한 목적의 DB를 용도별로 분류하는 것을 자동화하는 시스템에 관한 연구이다. 기존 수작업은 ERD를 기준정보로 전문가의 경험적 방법과 인터뷰로 수일간 임의분류를 수행한다. 본 연구는 DB의 SQL 이력 데이터를 전처리하여 군집분석 알고리즘을 활용해 수작업 특성분류를 대체하여 처리할 수 있음을 검증하였다. 시스템 얻는 시계열 데이터를 좌표계 데이터로 손쉽게 변환시키는 방안도 제시하였고, 결과 또한 실루엣 스코어를 통해 정확도 약 90.6%로 즉시 적용할 수 있음도 증명하였다.

향후 더 정확도 높은 분류 자동화를 위해 도메인 서비스를 제공하는 소스코드와 함께 자동화하는 분류연구가 추가로 필요하다.

References

[1] A. Mendelsohn, “The Oracle Story: 1984-2001,” *IEEE Annals of the History of Comput.*, vol. 35, pp. 10-20, Sep. 2012.

[2] B. J. Kim, “Verification of scalability and compatibility of TiDB for DBMS migration,” *J. Digital Contents Soc.*, vol. 23, no. 11, pp. 2299-2306, Nov. 2022.

[3] A. A. Hussein, “Data migration need, strategy, challenges, methodology, categories, risks, uses with cloud computing, and improvements in its using with cloud using suggested proposed model,” *J. Inf. Secur.*, vol. 12, no. 1, pp. 79-103, 19, Jan. 2021. (https://doi.org/10.4236/jis.2021.121004)

[4] M. H. G. Barbosa and P. H. M. Maia, “Towards identifying microservice candidates from business rules implemented in stored procedures,” *IEEE ICSA-C*, pp. 16-20, Mar. 2020. (https://doi.org/10.1109/ICSA-C50368.2020.00015)

[5] M. Stonebraker, A. Pavlo, R. Taft, and M. L. Brodie, “Enterprise database applications and the cloud: A difficult road ahead,” *IEEE Int. Conf. Cloud Eng.*, pp. 1-6, May 2014. (https://doi.org/10.1109/IC2E.2014.97)

[6] M. Ellison, R. Calinescu, and R. F. Paige, “Evaluating cloud database migration options using workload models,” *J. Cloud Comput.: Advances, Syst. and Appl.*, vol. 7, pp. 1-18, 2018. (https://doi.org/10.1186/s13677-018-0108-5)

[7] Z. Wang, W. Yan, and W. Wang, “Revisiting cloud migration: Strategies and methods,” *J. Physics: Conf. Series*, vol. 1575, no. 1, May 2020. (https://doi.org/10.1088/1742-6596/1575/1/012232)

[8] V. Velepucha and P. Flores, “A survey on microservices architecture: Principles, patterns and migration challenges,” *IEEE Access*, Aug. 2023.

[9] A. M. Ikotun, A. E. Ezugwu, L. Abualigah, B. Abuhaija, and J. Heming, “K-means clustering algorithms: A comprehensive review, variants analysis, and advances in the era of big data,” *Inf. Sci.*, vol. 622, pp. 178-210, Apr. 2023. (https://doi.org/10.1016/j.ins.2022.11.139)

[10] R. Mussabayev, N. Mladenovic, B. Jarboui, and R Mussabayev, “How to use K-means for big data clustering?,” *Pattern Recognition*, vol. 137, pp. 109-269, May 2023. (https://doi.org/10.1016/j.patcog.2022.109269)

[11] T. M. Kodinariya and R. Makwana, "Review on determining number of cluster in K-means clustering," *Int. J. Advance Res. in Comput. Sci. and Manag. Stud. Res. Paper*, vol. 1, pp. 90-95, Nov. 2013.

[12] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," *J. Computational and Applied Mathematics 20 (1987)*, vol. 20, pp. 53-65, Nov. 1987. (https://doi.org/10.1016/0377-0427(87)90125-7)

박 선 철 (Sun-chul Park)

2022년 : 숭실대학교 공학석사
2023년~현재 : 숭실대학교 일반대학원 인공지능 IT융합학과 박사과정
<관심분야> 인공지능, 클라우드, 데이터센터, 정보시스템 분석 및 설계
[ORCID:0000-0003-1838-974X]

김 영 한 (Young-han Kim)

1984년 : 서울대학교 졸업
1986년 : 한국과학기술원 공학석사
1990년 : 한국과학기술원 공학박사
1994년~현재 : 숭실대학교 전자정보공학부 교수
<관심분야> 차세대 인터넷 프로토콜, 이동/무선 네트워크, 인터넷 텔레포니, 센서/모바일 애드혹 네트워크