K-평균 군집화의 역사

K-평균 군집화의 개념은 1950년대 중반에 처음 등장했다. 1967년 제임스 맥퀸(James MacQueen)이 “K-평균”이라는 명칭을 사용했지만, 그 아이디어는 1956년 폴란드 수학자 **후고 스타인하우스(Hugo Steinhaus)**가 처음 제안했다 [1]. 이후 벨 연구소의 미국 물리학자 **스튜어트 로이드(Stuart Lloyd)**는 1957년에 해당 아이디어를 확장하여 실제 알고리즘을 고안했으나, 이 내용은 1982년에야 논문으로 출판되었다 [2] [1]. 1965년에는 미국 통계학자 **에드워드 포기(Edward W. Forgy)**도 유사한 방법을 발표하여, 이 알고리즘은 때때로 ‘로이드-포기 알고리즘’으로도 불린다 [1].

폴란드의 수학자 후고 스타인하우스(1887–1972). 그는 1956년 군집화 개념을 제안하여 K-평균 알고리즘의 기초를 마련했다 [1]. 이외에도 벨 연구소의 스튜어트 로이드(1957년 제안), 미국 통계학자 에드워드 포기(1965년 발표), 그리고 제임스 맥퀸(1967년 “K-평균” 명칭 사용) 등이 이 알고리즘의 발전에 중요한 기여를 했다 [1] [2].

K-평균 알고리즘의 원리

K-평균 군집화는 **비지도 학습**의 대표적인 방법으로, 데이터를 K개의 그룹으로 나누어 각 그룹 내 점들 간의 거리를 최소화한다. 즉, 데이터를 k개의 클러스터로 분할하여 각 데이터가 가장 가까운 클러스터 중심(평균)에 속하도록 반복적으로 조정한다 [3]. 주요 단계는 다음과 같다:

- **초기 중심 설정(Initialization)**: 임의로 K개의 중심점(centroid)을 선택한다 [4]. 예를 들어, 아래 그림은 $k = 3$ 인 경우 데이터 영역 내에 3개의 중심을 무작위로 배치한 모습이다.
  초기 단계: $k = 3$ 일 때 데이터 포인트(회색 점)와 임의로 선택된 3개의 중심점(색 점) [4].
- **군집 할당(Assignment)**: 각 데이터 포인트를 현재 중심 중 가장 가까운 중심에 할당한다. 거리는 주로 **유클리드 거리**를 사용하며, 즉 두 점 $\mathbf{x}$ 와 $\boldsymbol{\mu}$ 간의 거리는 $\sqrt{(x_1 - \mu_1)^2 + \cdots + (x_n - \mu_n)^2}$ 로 계산된다. 아래 그림은 각 점이 가장 가까운 중심(붉은·초록·파랑색)으로 할당된 모습이다 [5].
  할당 단계: 각 데이터가 가장 가까운 중심(색이 칠해진 점)으로 묶여 클러스터가 형성된 모습 [5].
- **중심 재계산(Update)**: 각 클러스터에 속한 데이터 포인트들의 평균 위치를 계산하여 중심을 이동시킨다. 새로운 중심 $\mu_i^{(t+1)}$ 는 해당 클러스터 $S_i^{(t)}$ 에 속한 점들의 좌표 평균으로 구해진다:

$$\mu_i^{(t+1)} = \frac{1}{|S_i^{(t)}|} \sum_{\mathbf{x}_j \in S_i^{(t)}} \mathbf{x}_j \quad \text{【3†L458 – L461】}.$$

  아래 그림은 클러스터 내부 점들의 평균 위치로 중심이 이동된 예시이다 [7]. 알고리즘은 이 할당 단계와 중심 재계산 단계를 중심이 더 이상 이동하지 않을 때까지 반복 수행한다(즉, 군집 할당이 바뀌지 않거나 WCSS(클러스터 내 제곱합)가 안정화될 때 종료) [8] [7].
  업데이트 단계: 각 클러스터의 중심(검은 점)이 클러스터 내 점들의 평균 위치로 이동한 모습 [7] [6].

이처럼 K-평균 알고리즘은 간단한 반복 과정을 통해 군집을 개선하며, 계산적으로 효율적이다. 다만 초기 중심 위치에 따라 결과가 달라질 수 있으며, 지역 최적해에 수렴할 수도 있다 [8].

K-평균 군집화의 실용 사례

고객 세분화(Customer Segmentation): 기업들은 고객의 나이, 소득, 구매 이력 등 특성 데이터를 K-평균으로 군집화하여 비슷한 성향의 고객 그룹을 식별한다 [9]. 이를 통해 각 그룹(예: 고소득층, 저소득 중간 사용층, VIP 고객 등)에

맞는 맞춤형 마케팅 전략을 수립할 수 있다. 예를 들어, 연구에 따르면 K-평균은 **비지도 학습**을 이용해 고객 하위 집단을 자동으로 찾아내고, 이를 통해 마케팅 타겟팅 효율을 높일 수 있음을 보여준다 [9].

아이리스 데이터셋 예시: 아이리스 데이터셋은 1936년 로널드 피셔가 소개한 대표적인 다변량 데이터로, 붓꽃 세 종(세토사·버지니카·버시컬러) 각 50개씩, 총 150개 샘플에 대해 꽃받침 길이/너비와 꽃잎 길이/너비 등 4개 특성이 측정되어 있다 [10] [11]. K-평균을 이용해 이 데이터를 $k = 3$ 으로 군집화하면, 아래 그림처럼 실제 종 분류와 유사한 패턴으로 데이터가 나뉘는 것을 볼 수 있다. 즉, K-평균은 비지도학습임에도 불구하고 종(Setosa 등)에 따른 클러스터를 대체로 잘 찾아낸다.
아이리스 데이터셋의 K-평균 군집화 예시: 3차원 산점도에서 3개의 클러스터(k=3)를 그린 결과(각 색은 서로 다른 클러스터) [10] [11]. 왼쪽 위는 클러스터 수 $k = 8$ 일 때, 오른쪽 위는 $k = 3$ 일 때 결과이며, 오른쪽 아래는 실제 정답(붓꽃 종)을 나타낸다. $k = 3$ 인 경우 K-평균 결과가 실제 종 분포와 비교적 유사함을 알 수 있다.

최신 기업 사례: 최근에는 실무에서도 K-평균 군집화가 널리 활용된다. 예를 들어, Airbnb 기술팀은 호스트의 숙소 이용 가능 패턴 데이터를 기반으로 K-평균을 적용하여 8개의 유형으로 호스트를 세분화했다 [12]. 이들은 k 값을 2에서 10까지 테스트한 후 엘보우 기법을 이용하여 $k = 8$ 이 최적임을 판단했으며 [12], 각 클러스터를 이용해 호스트 맞춤형 지원 서비스를 제공하고 있다. 유사하게, 우버(Uber)는 뉴욕시 차량 픽업 위치 데이터를 K-평균으로 클러스터링하여 **픽업 허브**(clusters centroid)를 찾았다 [13]. 이를 통해 운전자에게 특정 중심 지역 주변에서 대기하도록 안내하여 승객 대기 시간을 최소화했으며, 수요·공급 비율에 따라 동적인 요금 책정(서지 프라이싱)에도 활용하고 있다 [14]. 이러한 사례들은 K-평균을 통해 복잡한 실세계 데이터를 효과적으로 분할하여 비즈니스 문제(마케팅 전략, 운영 최적화 등)를 해결한 예들이다 [12] [13].

참고문헌: K-평균 군집화의 개념과 역사 [1] [2], 알고리즘 원리 [5] [6], 실용사례 [9] [10] [12] [13] 등을 참고하였다.

---

[1] [4] [5] [6] [7] [8] k-means clustering - Wikipedia
https://en.wikipedia.org/wiki/K-means_clustering

[2] K-Means Clustering for Machine Learning Explained
https://www.deeplearning.ai/the-batch/k-means-clustering-group-think/

[3] K-Means Clustering Explained
https://neptune.ai/blog/k-means-clustering

[9] Customer Segmentation using KMeans in R | GeeksforGeeks
https://www.geeksforgeeks.org/customer-segmentation-using-kmeans-in-r/

[10] [11] Iris flower data set - Wikipedia
https://en.wikipedia.org/wiki/Iris_flower_data_set

[12] From Data to Insights: Segmenting Airbnb's Supply | by Alexandre Salama | The Airbnb Tech Blog | Medium
https://medium.com/airbnb-engineering/from-data-to-insights-segmenting-airbnbs-supply-c88aa2bb9399

[13] [14] Uber trip segmentation using K-means clustering
https://www.linkedin.com/pulse/uber-trip-segmentation-using-k-means-clustering-khatre-csm-pmp