기초 이론부터 실무 실습까지

머신 러닝 익히기

programmers

Part 07. K-means Clustering

정 정 민

programmers

Chapter 17. K-means 실습

1. Mall Customers 데이터
2. EDA 및 전처리
3. 군집화 진행 및 결과 확인

Mall Customers 데이터

Mall Customers 데이터

- 이번 실습에서 사용할 데이터로 Kaggle의 공개 데이터 (링크)
  - 다운로드 받아주세요!

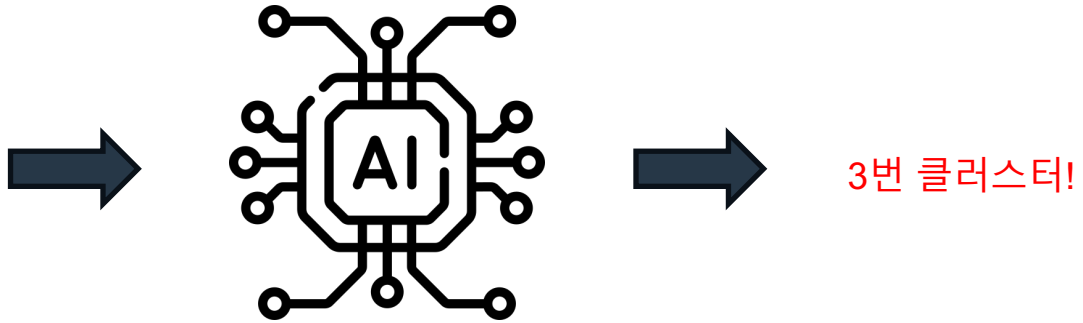
- 200명의 쇼핑몰 고객에 대한 정보 데이터
- 아래의 변수를 포함
  - 고객 ID
  - 성별
  - 나이
  - 연간 소득
  - 쇼핑 점수
    - 쇼핑 행동과 지출 성향을 기반으로 쇼핑몰에서 점수 부여

문제 정의

- 풀어야 하는 문제
  - 주어진 고객 데이터를 바탕으로 고객을 세분화(Customer Segmentation) 군집화
    독립 변수

- 머신 러닝 모델의 입, 출력 정의
  - 입력 : 고객 정보 데이터
  - 출력 : 클러스터링 결과

나이
성별
소득
쇼핑 점수



3번 클러스터!

- 수강생 여러분들이 실제 일하게 되실 업무 환경에서는 고객 세분화로 끝나는게 아니라
- “타겟 마케팅 전략 수립, 신규 고객 유치 방법, 재고 관리 최적화”와 같은
- 더욱 깊이 있는 문제를 설계 혹은 그것을 위한 방법으로 이런 과정이 사용됩니다.

EDA 및 전처리

강의 실습 코드 : 링크

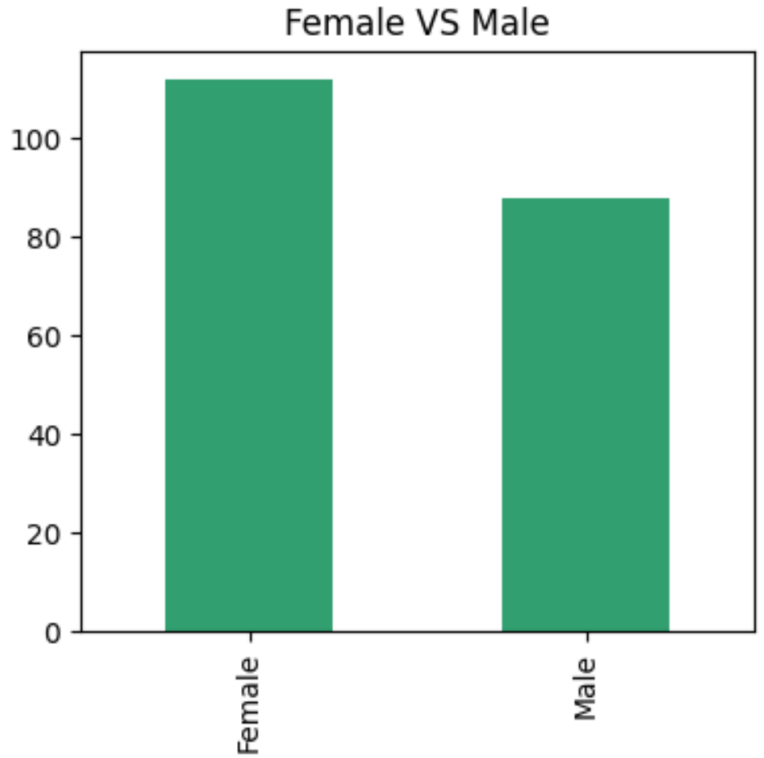
데이터 타입 및 고려사항

- **고객 ID**
  - 수치형 데이터
  - 고유 식별자로 일반적으로 학습에서 제외
- **성별**
  - 범주형 (카테고리형) 데이터
  - 인코딩 과정이 전처리로 필요함
- **나이, 연간 소득, 쇼핑 점수**
  - 수치형 데이터
  - 각자 서로 다른 스케일을 갖고 있음
  - 거리 기반 군집화에서 스케일 매칭이 매우 중요한 요소

성별 시각화

- 쇼핑 데이터이므로 약~간의 치우침이 있지만 큰 차이로 보이지는 않음
- 즉, 성별 분포는 비슷한 정도의 수준

- 치우침으로 인한 추가 전처리는 고려하지 않아도 될 것 같고
- 오히려 이런 약간의 치우침이 데이터의 특성을 잘 보여주고 있음



성별 인코딩

- 원-핫 인코딩이 2개의 범주에 적용되면 이진 인코딩이라고 부를 수 있음

- 대신, 원-핫 인코딩은 column의 이름이 바뀐다는 점이 다르고
  - ex) 성별_여성 : 1 또는 0
- 이진 인코딩은 1이 어떤 값이며 0이 어떤 값인지를 숙지하고 있어야 함
  - ex) 성별 : 1 또는 0
  - 1은 여성 & 0은 남성

- 이번에는 이진 인코딩을 사용
  - 여성(Female)을 1로, 남성(Male)을 0으로 매핑

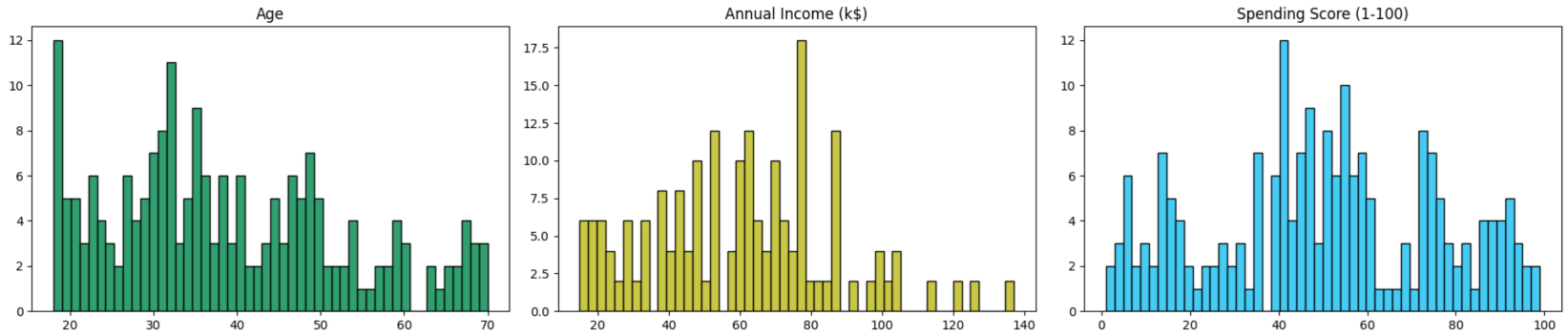
```
# 성별을 이진 변수로 변환
categori_data_encode = pd.DataFrame(
  customers['Genre'].replace({'Female': 1, 'Male': 0})
)
categori_data_encode.columns = ['Gender']
categori_data_encode
```

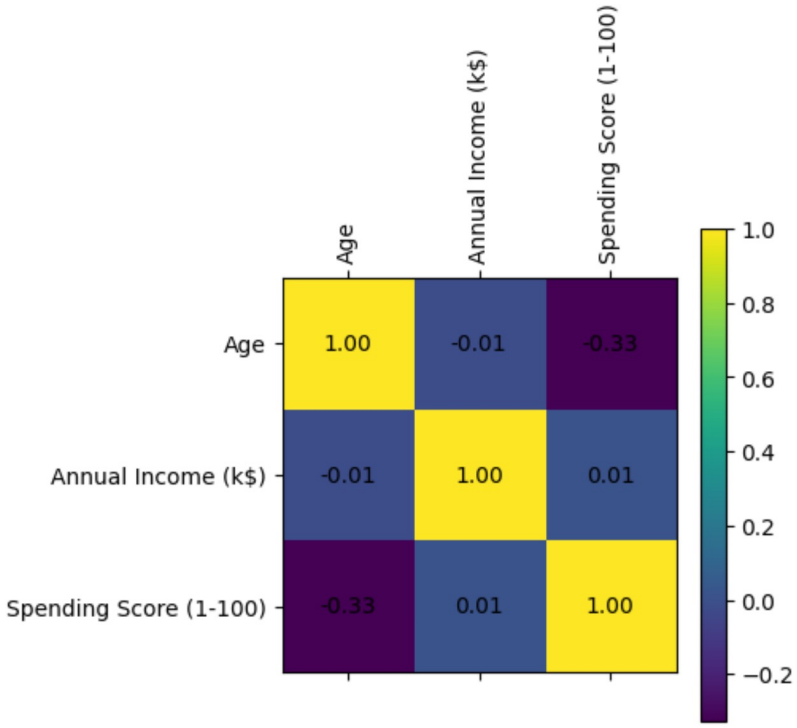
수치형 데이터 시각화

- 나이와 소득은 인구 통계학에 근거한 데이터에서 크게 벗어나지 않음
- 따라서 어느 정도 정규 분포의 분포를 예측할 수 있고 실제 결과도 그러함
- 쇼핑 점수도 정규 분포를 어느 정도 모방함

- 소득 특성은 아웃라이어가 발생하기 가장 쉬운 데이터임에도
  제거가 필요해 보이는 이상치까지는 보이지 않음

- 상관 관계 분석에서도 세 변수가 큰 상관관계를 갖고 있지 않음





수치형 데이터 정규화

- 3개의 변수를 정규 분포로 가정할 수 있으므로
- Min-Max 스케일링 보다는 Standard 스케일링이 더 적합해 보임

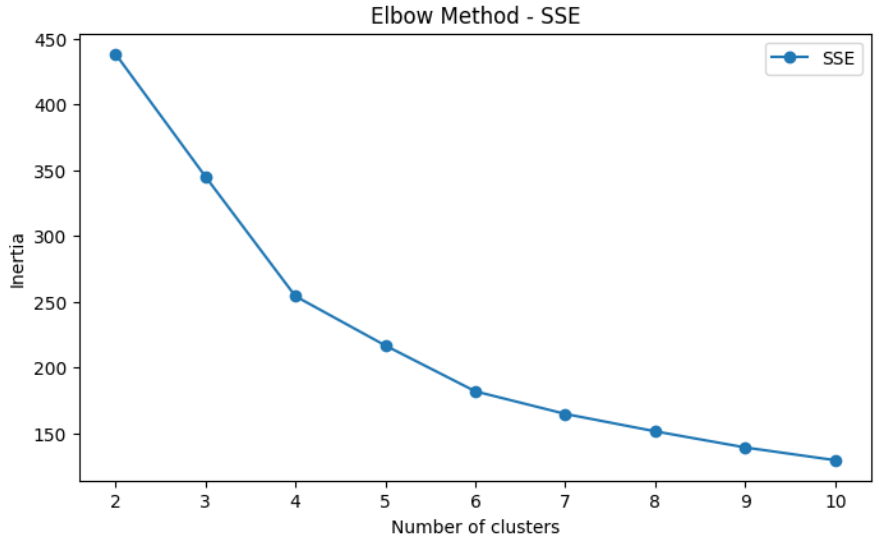
```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()

# 수치형 데이터 스케일링
numeric_data = customers[numeric_columns]
numeric_data_scaled = scaler.fit_transform(numeric_data)
numeric_data_scaled = pd.DataFrame(numeric_data_scaled)
numeric_data_scaled.columns = numeric_columns
```

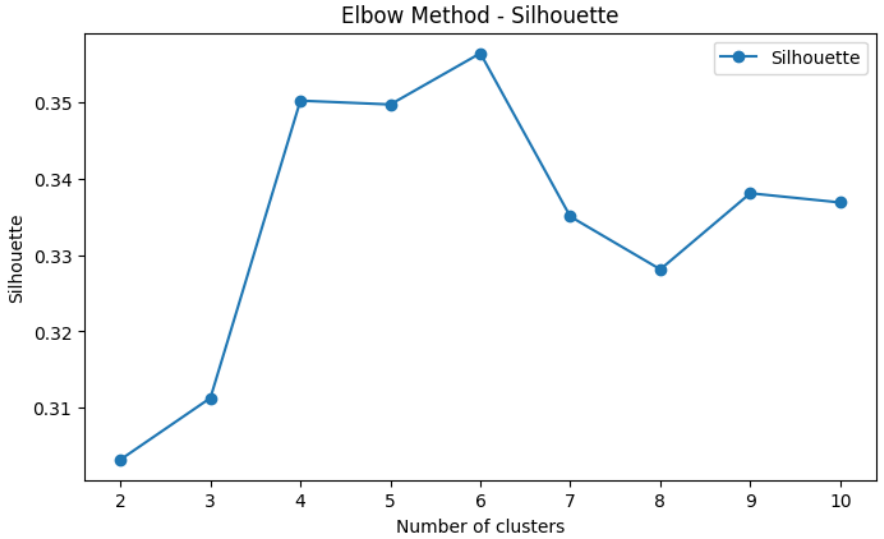
군집화 진행 및 결과 확인

강의 실습 코드 : 링크

최적 K 값 찾기

- K 값의 변화에 따른 SSE 와 실루엣 계수를 확인
- SSE 에서는 점진적으로 값의 변화가 있음
  - 그나마 K=4 혹은 K=6 에서 감소율 변화가 보임
- 실루엣 계수는 K=6에서 최고 값을 갖음
- 최적 K=6으로 선택
- 실루엣 계수의 변동이 있는 것은 매우 일반적인 현상
  - 데이터 구조의 복잡성
  - 잡음과 이상치
  - 균일하지 않은 밀도 & 데이터 간 거리
- 실루엣 계수 간 차이가 크지 않다면 작은 K를 기준으로 하는게 좋은 선





학습 결과 확인

- SSE와 Silhouette Coefficient 값을 이용해 정량 평가가 가능
- 사실 이미 최적 K를 찾는 과정에서 확인하긴 함!

- 실루엣 계수가 0.36 정도라면 어느 정도 군집화가 의미 있게 진행 됨
- 하지만 계선의 여지는 있을 수 있음

- 이를 직관적으로 보기 위해서는 다음장의 시각화 과정이 필요함

```
y_pred = kmeans.predict(customers_combined)
silhouette_avg = silhouette_score(customers_combined, y_pred)

print("SSE Value : {:.2f}".format(kmeans.inertia_))
print("Silhouette Score: {:.2f}".format(silhouette_avg))

# SSE Value : 181.95
# Silhouette Score: 0.36
```

결과 시각화

- 학습 결과로 나온 클러스터를 이용해 시각화를 할 수 있음

- 일반적으로 사용한 데이터는 2차원 이상이므로
- PCA 혹은 T-SNE 등의 방식을 활용해 2차원으로 차원 축소 가능

- 각각의 feature 축이 어떠한 의미인지는 알기 어렵지만
- 결과적으로 Cluster가 잘 생성된 것을 확인!

각 클러스터 분석

- 클러스터 생성을 위한 목적만이라면 앞장의 내용으로도 충분하지만
- 일반적으로는 만들어진 클러스터가 어떤 의미가 있는지 도메인 지식을 이용해 추측해야 함

- 원래 데이터의 형태로 전처리의 역과정을 거쳐 데이터의 재건한 뒤
- 각 클러스터에 포함된 데이터의 의미를 확인해야 함
  - 이때 정해진 방법이 있는 건 아닙니다.
  - 간단하게 기본 정보 혹은 기술 통계를 보거나
  - 상관관계 분석, 다른 머신 러닝 모델 생성 등의 과정이 필요합니다.

	Cluster	0	1	2	3	4	5
Age	count	38.000000	39.000000	23.000000	45.000000	21.000000	34.000000
	mean	27.000000	32.692308	25.000000	56.333333	45.523810	41.264706
	std	7.032742	3.728650	5.300086	8.453079	11.766984	10.768385
	min	18.000000	27.000000	18.000000	43.000000	20.000000	19.000000
	25%	21.000000	30.000000	21.000000	49.000000	36.000000	34.500000
	50%	26.500000	32.000000	23.000000	54.000000	46.000000	42.500000
	75%	31.750000	35.500000	29.500000	65.000000	53.000000	47.000000
	max	40.000000	40.000000	35.000000	70.000000	67.000000	59.000000

E.O.D