**제목: KMeans++ 및 TF-IDF를 사용한 유튜브 댓글의 스팸 탐지**

**Abstract**

본 프로젝트는 YouTube 댓글에서 스팸 댓글을 식별하고 클러스터링하는 포괄적인 데이터 분석 방법론을 제시합니다. 주요 목표는 스팸 댓글을 올바르게 필터링하여 사용자 경험을 향상시키고 콘텐츠 제작자의 신뢰성을 유지하는 것입니다. 인기 있는 YouTube 동영상에서 댓글을 주요 데이터로 수집하고, 세심한 데이터 전처리를 통해 분석을 위한 준비 단계를 거쳤습니다.

Feature Extraction단계에서는 TF-IDF 벡터화를 활용하였고, 이를 통해 중요한 특징 벡터를 생성하였습니다. 데이터의 구조와 스팸 식별에 대한 다양한 관점을 제공하기 위해 KMeans++, 계층적 클러스터링, DBSCAN 등의 클러스터링 기법을 적용하였습니다. PCA를 사용한 차원 축소 및 t-SNE 시각화를 통해 데이터의 복잡성을 줄이면서도 주요 분산을 유지하였습니다.

클러스터링의 질은 Silhouette 지수, Davis-Boldin 지수, Kalinsky-Haravaz 지수 등을 활용하여 평가하였습니다. 이를 통해 클러스터링 매개변수와 feature 수의 최적화를 도와 스팸 댓글 탐지의 효율성을 높였습니다. 분석 결과, TF-IDF 벡터화 매개변수의 조정과 차원 축소와 feature 선택 사이의 균형이 스팸 댓글 탐지의 효율성을 극대화하는 데 중요하다는 것을 확인하였습니다.

해당 프로젝트는 스팸 댓글을 성공적으로 클러스터링하고 분류하는 방법을 제시하였으나, 특정 데이터세트에 대한 과적합 가능성과 신규 댓글 분류에 대한 추가 연구 필요성 등의 한계를 인식하고 있습니다. 그럼에도 불구하고 이 연구는 스팸 댓글 문제를 해결하는 방법에 대한 데이터 과학의 가능성을 보여주며, 온라인 콘텐츠 관리와 사용자 상호 작용 공간 개선에 대한 새로운 틀을 제공합니다.

**Introduction**

YouTube는 대중적인 커뮤니티 플랫폼으로, 사용자 간의 댓글을 통한 상호 작용이 활발하게 이루어집니다. 그러나 이러한 상호작용이 증가함에 따라, 스팸 댓글 문제가 심각해지고 있습니다. 스팸 댓글은 사용자의 경험을 해치고 콘텐츠 제작자의 신뢰성을 저하시키므로, 이를 효과적으로 관리하는 것이 중요합니다.

이에 KMeans++ 클러스터링과 TF-IDF 벡터화를 활용하여 스팸 댓글을 식별하고 클러스터링하는 방법을 제시합니다. KMeans++는 기존 KMeans 알고리즘의 향상된 버전으로, 더욱 효과적인 클러스터링 결과를 제공합니다. TF-IDF는 텍스트 데이터를 숫자로 변환하여 알고리즘에 적용할 수 있게 하는 핵심 기술입니다.

본 프로젝트를 통해, 스팸 댓글 문제를 해결하고, 온라인 콘텐츠와 사용자 상호 작용 환경을 개선하는 데 필요한 데이터 과학의 가능성을 보여줍니다. 또한, YouTube의 스팸 댓글 특성에 대한 이해를 높이고, 스팸의 영향을 완화하는 효과적인 해결책을 제시합니다.

**Methodology**

1. **Data Collection**

꼼꼼한 데이터 수집 절차를 통해 스팸 댓글 탐지에 효과적인 데이터 세트를 확보하였습니다. 저희 조는 인기 있는 YouTube 동영상에 초점을 맞추었는데 해당 동영상들이 큰 가시성과 트래픽을 얻을 가능성이 높아, 욕설이나 광고 등 스팸 댓글이 많이 등장할 것이라는 가설을 바탕으로 한 것입니다. 댓글과 함께 동영상 메타데이터에 접근하기 위해서는 YouTube의 데이터 API를 활용하였습니다. 댓글에 달린 대댓글은 영상과 관련성이 적거나 댓글에 대한 반응이 주를 이루는 경우가 많아, 대댓글은 수집 대상에서 제외했습니다.

수집한 데이터는 'id'와 'comment'라는 두 가지 주요 열로 구성되었습니다. 'id'는 각 댓글의 고유 식별자로, 추적 및 개별 분석에 중요한 역할을 합니다. 'comment' 열에는 사용자 댓글의 실제 텍스트가 포함되어 있습니다.

데이터 수집 과정에서는 사용자의 개인 정보를 존중하고, 공개적으로 사용 가능한 데이터만을 이용하도록 윤리적 가이드라인을 준수했습니다. 인기 있는 YouTube 동영상 중에서 몇 개를 선택해 댓글 데이터를 성공적으로 수집했습니다. 이렇게 수집된 데이터는 저희가 설정한 전처리 과정을 거쳐, 클러스터링 및 스팸 감지 작업을 준비하였습니다. 이러한 과정을 통해 데이터 분석을 위한 기초를 마련하였습니다.

1. **Data Preprocessing**

데이터 전처리 단계가 후속 분석을 위한 YouTube 댓글 데이터를 준비하는 데 매우 중요한 역할을 하였습니다. 이 과정에서는 데이터를 정리하고 표준화하여 클러스터링 알고리즘이 관련 텍스트 콘텐츠에만 초점을 맞추도록 보장하였습니다.

첫 번째로, 데이터 세트의 일관성을 유지하기 위해 한국어가 아닌 댓글을 제외하였습니다. 여러 언어로 작성된 댓글이 섞여 있을 경우 분석에 불필요한 변동성이 생겨, 클러스터링 성능이 저하될 수 있기 때문입니다. 한국어가 아닌 텍스트는 한글 문자를 식별하여 제거하는 정규식(regex) 패턴을 활용하였습니다.

다음 단계에서는 숫자와 특수 문자를 댓글에서 제거하였습니다. 유튜브 댓글에는 영상의 특정 부분을 지정하는 재생 시간이 포함되는 경우가 많지만, 이런 숫자들이 댓글이 스팸인지 아닌지를 판단하는 데 큰 영향을 미치지 않는다고 판단하여 모두 삭제하였습니다. 또한 스팸 댓글에서는 특수 문자를 과도하게 사용하여 글의 가독성을 해치고, 텍스트 변환 과정에 방해가 될 수 있다고 판단하여 이러한 특수 문자들도 제거하였습니다. 이렇게 함으로써 스팸 필터링의 정확성을 향상시킬 수 있었고 이 과정은 pandas의 정규식 기능을 활용하여 진행하였습니다.

또한, 'ㅋㅋㅋㅋ'나 'ㅎㅎㅎㅎ'와 같이 웃음이나 감정을 표현하는 한국어 자음만으로 이루어진 단어를 제거하였습니다. 이런 문자 시퀀스는 특정 사회적 맥락에서는 의미가 있지만, 스팸 탐지에는 의미가 없으며, 분석 결과를 왜곡할 수 있기 때문입니다.

이러한 전처리 결과, 불필요한 숫자와 기호, 관련 없는 문자 시퀀스를 제거한 깔끔한 데이터 세트를 얻을 수 있었습니다. 이렇게 정제된 데이터 세트는 특징 추출 단계를 위해 준비되었으며, 기계 학습 알고리즘이 처리할 수 있는 형식으로 텍스트를 변환하였습니다.

1. **Feature Extraction**

분석을 위해 TF-IDF 기능 값을 최적화하는 과정에서 TfidfVectorizer 내에서 여러 하이퍼 매개변수가 꼼꼼하게 조정되었습니다. 이러한 조정은 YouTube 댓글 데이터에서 파생된 특징 벡터의 질을 개선하는 데 매우 중요했습니다.

먼저 'max\_features' 매개변수를 60으로 설정했습니다. 특징 벡터의 용어 수에 대한 이 특정 제한은 여러 시도를 거쳐 선정되었습니다. 목표는 가장 적절하고 관련성이 높은 용어를 정확하게 포착하는 동시에 노이즈나 덜 중요한 정보가 포함되는 것을 방지하는 최적의 범위를 식별하는 것이었습니다. 이러한 신중한 선택을 통해 스팸 콘텐츠를 잠재적으로 나타낼 수 있는 용어에 초점을 맞춰 기능 공간을 관리 가능하고 의미 있게 만들 수 있습니다.

우리가 조정한 또 다른 주요 하이퍼파라미터는 0.6으로 설정된 'max\_df'였습니다. 즉, 문서의 60% 이상(이 경우 YouTube 댓글)에 나타나는 모든 용어는 feature 벡터에서 제외됩니다. 이 결정은 전체 데이터 세트에서 너무나 흔하게 나타나는 단어들이 개별 댓글을 구별하는데 큰 도움이 되지 않는다는 이해를 바탕으로 합니다. 특히 스팸 탐지에서는 이런 일반적인 단어들이 텍스트에서 독특하거나 의심스러운 패턴을 찾아내는데 크게 기여하지 못할 수 있습니다.

ngram\_range는 (1,2)로 설정하여, 유니그램(단일 단어)과 바이그램(두 단어의 조합)을 모두 고려하도록 하였습니다. 바이그램을 포함시키면, 스팸 댓글의 특성이 될 수 있는 특정한 문구나 일반적인 단어 쌍을 찾아낼 수 있습니다. 이는 유니그램만을 사용하는 것보다 더욱 세밀한 분석을 가능하게 하며, 이를 통해 스팸 댓글을 더 정확하게 구별할 수 있습니다.

1. **Data Normalization**

스팸 댓글 감지를 위한 YouTube 댓글 분석에서는 Data Normalization은 중요한 단계입니다. 이는 효과적인 클러스터링에 필요한 데이터를 준비하는 핵심 과정입니다.

댓글을 벡터화할 때 TF-IDF를 사용하였는데 이 벡터들은 크기가 분석 결과를 왜곡하지 않게 하기 위해 정규화가 필요하며, 이를 위해 유클리드 정규화 또는 L2 정규화라는 기법을 사용했습니다.

정규화는 모든 댓글을 동일한 기준으로 취급하기 위해 필요하며, 이를 통해 댓글의 길이가 분석에 불균형적인 영향을 미치는 것을 방지합니다. 또한, 이는 클러스터링 알고리즘의 성능을 향상시키고, 텍스트 유사성을 더 잘 포착하도록 돕습니다.

1. **Similarity Measurement**

YouTube 댓글 분석에서 댓글 간의 유사성 평가는 스팸 감지에 중요한 요소입니다. 이를 위해 Cosine Similarity라는 기법을 사용하는데, 특히 계층적 클러스터링과 DBSCAN과 같은 클러스터링 기술에 효과적입니다.

Cosine Similarity는 두 벡터 사이의 각도를 측정하여 유사성을 파악하는 방법으로, 댓글의 길이와 상관없이 내용의 유사성을 파악하는데 용이합니다. 계층적 클러스터링은 데이터 포인트를 그룹화하는데 Cosine Similarity를 효과적으로 사용하며, DBSCAN은 빽빽하게 모여 있는 데이터를 그룹화하고, 이상치를 식별하는데 Cosine Similarity가 유용합니다.

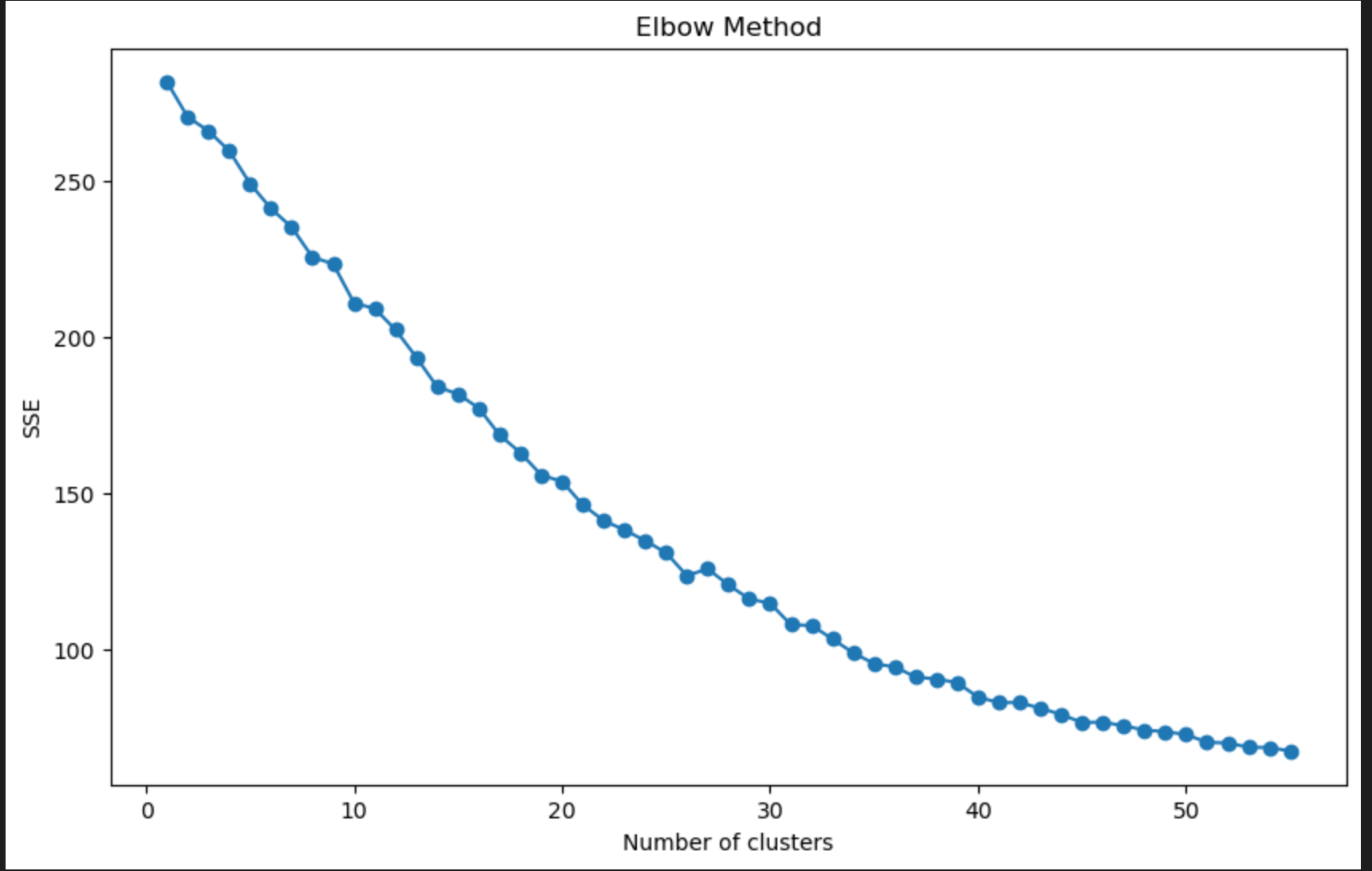
그러나 KMeans++와 같은 알고리즘에는 Cosine Similarity가 적합하지 않습니다. KMeans++는 클러스터가 구형이라는 가정을 하며, 각 클러스터의 중심을 계산하는 방식이 Cosine Similarity와 잘 맞지 않습니다. 또한, 고차원 공간에서의 효율성 문제도 있습니다.

결론적으로, Cosine Similarity는 텍스트 문서 간의 유사성을 평가하는 좋은 도구이지만, 적용하려는 클러스터링 기술에 따라 그 효과가 달라질 수 있습니다. 특히 KMeans++와 같은 알고리즘에서는 유클리드 거리와 같은 다른 유사성 측정 방법이 더 적합해 사용하지 않았습니다.[2]

1. **Clustering Analysis**

스팸 댓글 감지를 위한 YouTube 댓글 분석에서 클러스터링 분석은 중요한 역할을 합니다. 특히, 이 분석은 KMeans++ 클러스터링 알고리즘의 사용, 최적의 클러스터 수 결정, 그리고 결과의 시각화 등을 포함합니다.

KMeans++는 표준 KMeans 알고리즘의 향상된 버전으로, 서로 멀리 떨어진 초기 중심점을 스마트하게 선택해 더 안정적인 클러스터링 결과를 제공합니다. 이는 YouTube 댓글 데이터의 복잡성을 효과적으로 다루기 위해 필요한 강력한 클러스터링 방법을 제공합니다.[3]



최적의 클러스터 수를 결정하기 위해 Elbow method를 사용하였습니다. 이 방법은 다양한 클러스터 수에 대해 KMeans++ 알고리즘을 실행하고 각각의 제곱 오차 합(SSE)를 계산하여, SSE 값이 크게 변화하지 않는 지점을 찾아 최적의 클러스터 수를 결정합니다.

Elbow Method를 활용해 최적화된 cluster의 갯수를 찾으려고 했지만 , 완만해지는 부분이 존재하지 않아 cluster의 갯수를 정하는데 활용하지는 못했습니다. 이는 댓글들 중 하나로 clustering될 수 있는게 다수 존재해 SSE가 계속해서 낮아질 수 있기 때문으로 보입니다. 따라서 clustering 되는 결과를 확인하고 클러스터의 개수를 결정했습니다.[4]

1. **Alternative Clustering Techniques**

KMeans++ 외에도 계층적 클러스터링과 DBSCAN 등의 대체 클러스터링 기술을 활용하였습니다. 이러한 다양한 방법은 데이터에 대한 광범위한 관점을 제공하고 분석 결과를 검증하는데 도움이 됩니다.

계층적 클러스터링은 데이터를 연속적으로 병합하거나 분할하여 클러스터의 계층 구조를 형성하는 방법입니다. 이 방법은 댓글 데이터의 중첩된 구조를 파악하고 자연스러운 그룹화에 대한 통찰력을 제공하는 데 유용하였습니다. 이를 구현하기 위해 'AgglomerativeClustering' 클래스를 사용하였으며, 클러스터링은 Cosine similarity matrix를 기반으로 적용되었습니다.[5]

DBSCAN은 밀도 기반의 클러스터링 기법으로, 저밀도 영역의 이상치를 효과적으로 식별할 수 있습니다. 이는 스팸 댓글과 같이 일반적인 패턴을 따르지 않는 이상치를 탐지하는 데 특히 유용하였습니다. DBSCAN 역시 Cosine similarity matrix를 기반으로 적용되었습니다.[6]

1. **Dimensionality Reduction**

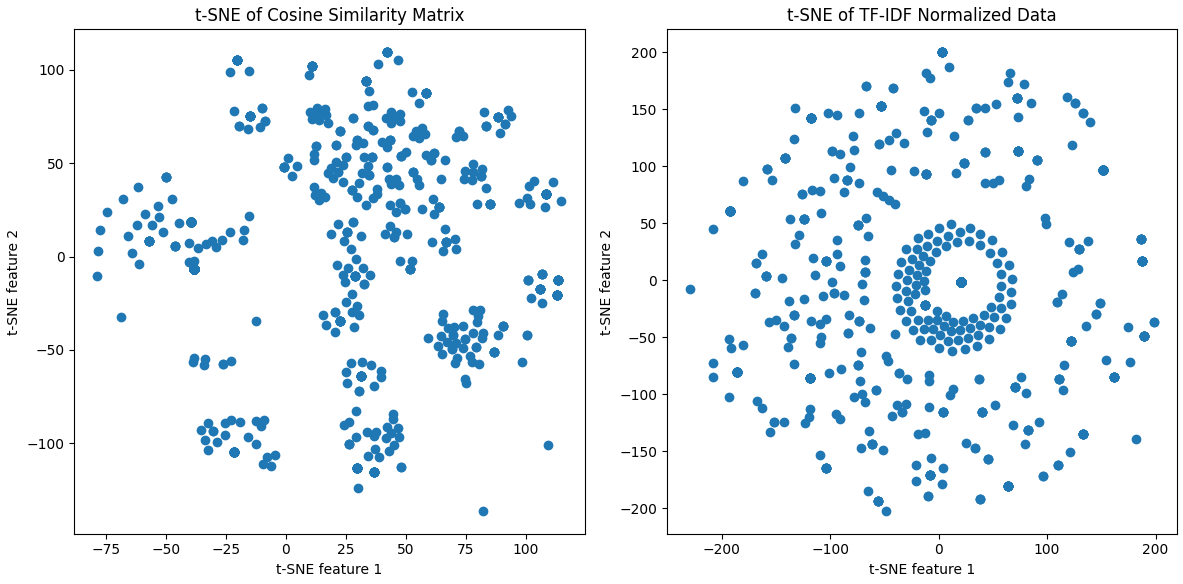
텍스트 데이터와 같은 고차원 데이터를 처리하는데, PCA(주성분 분석)는 핵심적인 단계입니다. TF-IDF 벡터화를 통해 고차원 공간으로 변환된 데이터에 PCA를 적용하여, 중요한 변동성을 가진 feature를 캡처하고 압축하였습니다. 이 과정은 계산 복잡성을 줄이는 동시에, 가장 구별력 있는 특성에 집중하여 클러스터링 결과의 품질을 향상시키는 역할을 하였습니다. [7][8]

데이터의 총 분산 중 얼마나 많은 부분이 포착되는지를 시각적으로 나타내는 누적 설명 분산 그래프를 통해, 분석에 유지할 주성분의 최적 수를 결정하였습니다. 데이터 분산의 95% 이상이 유지되도록 주성분 수를 선택하였는데, 이는 축소된 데이터 세트가 원래의 고차원 공간에 포함된 대부분의 정보를 여전히 포함하도록 하기 위함입니다. 이 결정은 데이터 세트를 단순화하면서도 정확성을 유지하였습니다.

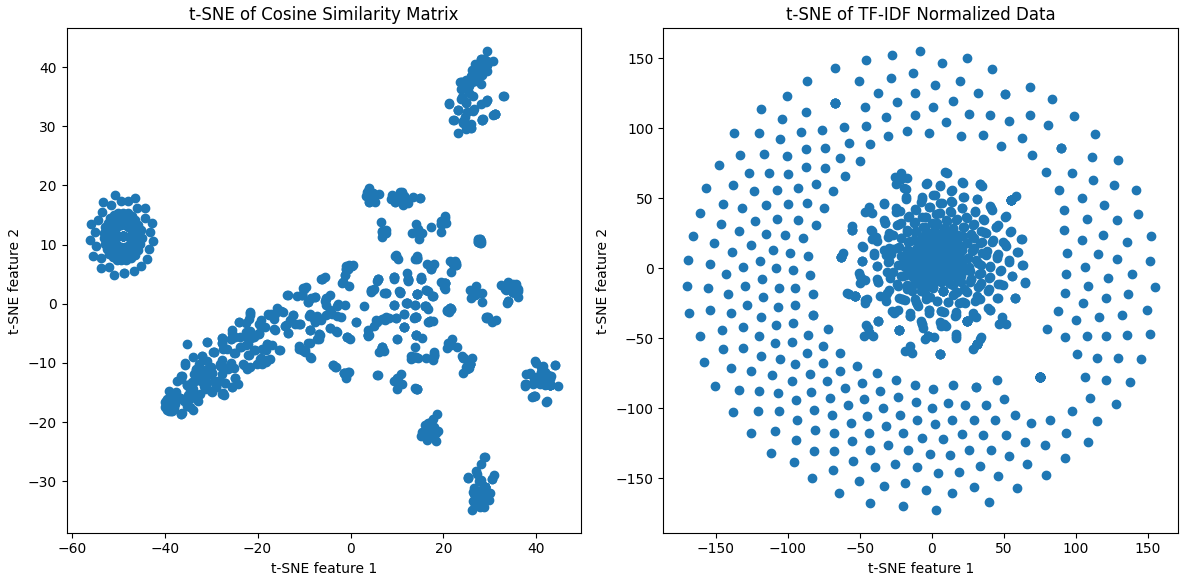
텍스트 데이터는 TF-IDF 과정을 통해 고차원 데이터로 생성하는데, 이를 시각화해서 클러스터링이 대략 어떻게 이루어질지 예상하는데 t-SNE를 사용하였습니다. 고차원의 데이터이기 때문에 t-SNE를 통해 2차원으로 표현한 모습이 고차원 데이터를 잘 표현한다고 하기 어렵지만 tf-idf의 파라미터를 조정하는 보조적 수단으로 활용하였습니다.[9]

tf-idf 진행 후 ,cosine similarity, normalized data에 대해 다음 모습과 같이 나타났습니다.

tfidf\_vectorizer = TfidfVectorizer(max\_df=0.6,ngram\_range=(1,2), max\_features=60, stop\_words='english')



tfidf\_vectorizer = TfidfVectorizer(max\_df=0.6,ngram\_range=(1,2), max\_features=1200, stop\_words='english')



두 결과를 비교했을 때 밑이 clustering 품질이 안 좋은 것을 확인할 수 있습니다. 다른 Clustering 평가요소와 함께 tf-idf의 파라미터값을 조정하는데 활용하였습니다.

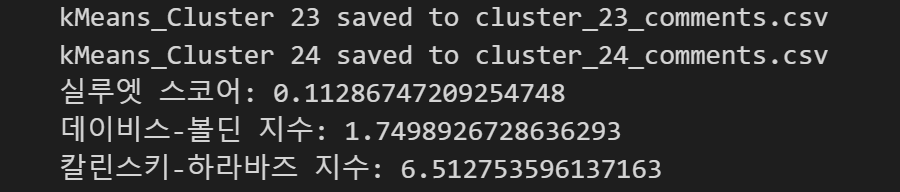
1. **Cluster Evaluation**

**Silhouette(실루엣) 지수(계수**)는 클러스터링 결과를 평가하는 지표로서 각 데이터별로 그 데이터가 속한 군 내의 유사도와 인접한 군의 유사도를 비교하는 지표입니다. 군 내 유사도가 군 간 유사도보다 큰 상황이면 클러스터링이 잘 되었다는 뜻인데 이는 -1과 1사이의 값에서 1에 가까울수록 잘 된 것입니다.[10]

**데이비스-볼딘 지수**는 클러스터 내의 산포도와 클러스터 간의 분리도를 통합적으로 고려한 지표입니다. 해당 지수는 분리도가 크고, 산포도가 작은 경우 작은 값을 가진다. 따라서 결과값이 작을수록 클러스터링이 잘 이루어졌다고 판단할 수 있습니다.[11]

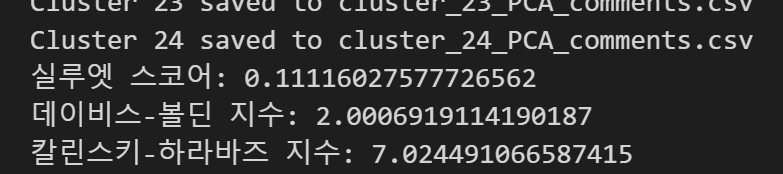
**칼린스키-하라바즈 지수**는 클러스터 간 분산과 클러스터 내 분산의 비율을 바탕으로 클러스터링의 품질을 평가합니다. 클러스터 간 분산은 클러스터 중심들이 전체 데이터의 중심으로부터 얼마나 멀리 떨어져 있는지를 측정하고, 클러스터 내 분산은 클러스터 내의 데이터 포인트들이 클러스터 중심으로부터 얼마나 멀리 떨어져 있는지를 측정합니다. 클러스터 간 분산이 커질수록, 클러스터 내 분산이 작을수록 값이 커지게 됩니다. 따라서 이 지수가 높을수록 클러스터링이 잘 이루어졌다고 판단할 수 있습니다.[12]

(tf\_idf,max=1200,ngram(1,2))

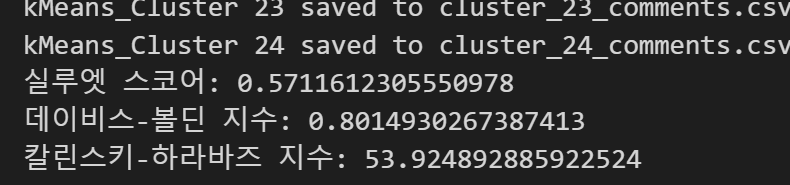


(tf\_idf,max=1200,ngram(1,2))

pca 1200->468 차원 축소후 kmeans

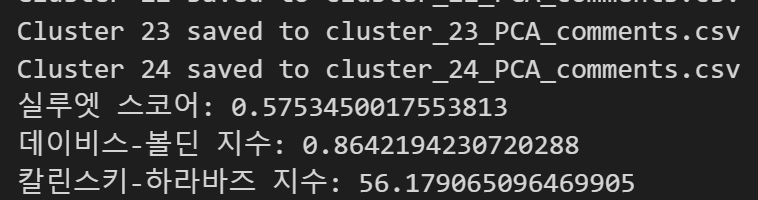


(tf\_idf,max=60,ngram(1,2))

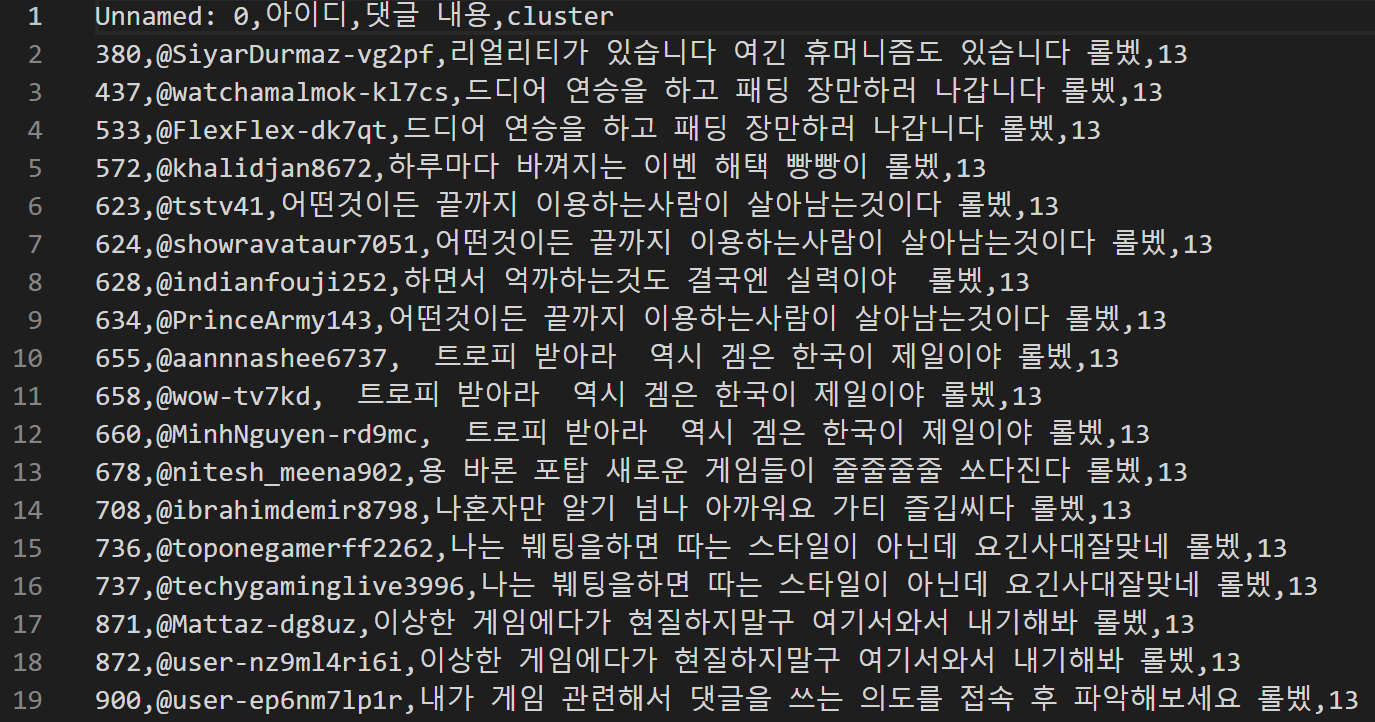


(tf\_idf,max=60,ngram(1,2))

pca 60->49



클러스터링 된 결과 예시1



분석된 YouTube 댓글 데이터세트에는 '롤벴'과 '롤뱄' 키워드가 각각 18개와 22개 포함된 40개의 스팸 댓글이 포함되어 있습니다. TF-IDF 벡터화를 통해 최대 1200개의 feature를 고려하고 유니그램과 바이그램을 모두 사용한 경우, KMeans++ 클러스터링 알고리즘은 이 중 절반인 20개의 스팸 댓글을 성공적으로 감지하였습니다.

그러나 주성분 분석(PCA)을 통해 feature space를 1200차원에서 468차원으로 줄이는 과정에서 스팸 탐지율이 하락하였습니다. 이를 위해 최대 feature 수를 60개로 줄이고 n\_gram 매개변수를 (1,2)로 유지한 결과, 스팸 댓글 37개를 성공적으로 감지할 수 있었습니다.

마지막으로, PCA를 다시 적용하여 feature 개수를 49개로 줄인 결과, 두 개의 클러스터에서 총 36개의 스팸 댓글을 감지하였습니다. 이 결과는 TF-IDF 벡터화 매개변수와 KMeans++ 클러스터링에 사용된 feature 수의 조정이 스팸 탐지 효율성에 큰 영향을 미친다는 것을 보여줍니다. 이를 통해 차원 축소와 feature 선택 사이의 균형이 스팸 탐지 성능을 최적화하는 데 중요함을 확인할 수 있습니다.

**Results and Conclusion.**

탐지 목표로 했던 스팸 댓글들 중 높은 확률로 클러스터링 하는데 성공했습니다. 원래 기존의 데이터에는 단어 사이에 특수기호가 중간에 포함되어 있었지만 전처리를 통해 제거하니 결과가 좋게 나온 것 같습니다. 하지만 한계점으로 영상마다 댓글의 수 , 단어의 수가 모두 다르다보니 파라미터가 데이터에 오버피팅될 가능성이 크다는 한계점이 존재합니다. 또한 단순히 스팸 댓글을 클러스터링하는데 그치다 보니 새로 작성된 댓글을 스팸댓글이라고 식별하거나 어느 클러스터에 속하는지 분류하는 것은 추가적인 연구가 필요합니다. 하지만 클러스터링된 결과를 가지고 스팸 댓글의 주요 키워드가 무엇인지는 뽑아낼 수 있었고 , 스팸댓글을 성공적으로 분류하는 데에는 성공했습니다. 또한 PCA를 통해 차원을 축소해도 클러스터링을 성공적으로 하였습니다.

**reference**

[1]“sklearn.feature\_extraction.text.TfidfVectorizer — scikit-learn 1.3.2 documentation,” Scikit-learn.org. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.feature\_extraction.text.TfidfVectorizer.html. [Accessed: Dec. 08, 2023].

[2]"Introduction to Information Retrieval," by C. Manning, P. Raghavan, and H. Schütze, Cambridge University Press. This book provides insights into text analysis techniques, including cosine similarity.

[3]"Data Mining: Concepts and Techniques," by J. Han, M. Kamber, and J. Pei, Morgan Kaufmann Publishers, which covers various clustering algorithms including KMeans and their properties.

[4]"Pattern Recognition and Machine Learning," by Christopher M. Bishop, Springer. This text provides insights into different techniques for model selection, including the Elbow Method in the context of clustering.

[5]"Introduction to Data Mining," by P.-N. Tan, M. Steinbach, and V. Kumar, Pearson Education. This book explains different clustering methods, including hierarchical clustering and its applications.

[6]"Mining of Massive Datasets," by J. Leskovec, A. Rajaraman, and J. D. Ullman, Cambridge University Press. It includes comprehensive coverage of density-based clustering techniques like DBSCAN.

[7]"Pattern Recognition and Machine Learning," by Christopher M. Bishop, Springer. This text offers insights into PCA and its role in dimensionality reduction.

[8]"The Elements of Statistical Learning: Data Mining, Inference, and Prediction," by T. Hastie, R. Tibshirani, and J. Friedman, Springer. This book covers various aspects of statistical learning, including the use of PCA for dimensionality reduction.

[9]"Visualizing Data using t-SNE," by L. van der Maaten and G. Hinton, Journal of Machine Learning Research. This paper introduces t-SNE and discusses its applications in data visualization.

[10]"Data Clustering: Theory, Algorithms, and Applications," by Guojun Gan, Chaoqun Ma, and Jianhong Wu, ASA-SIAM Series on Statistics and Applied Probability. This book provides comprehensive coverage of clustering methods, including an in-depth discussion on the Silhouette Index, its computation, and application in cluster evaluation.

[11]"Cluster Analysis: Basic Concepts and Algorithms," a chapter in a data mining or machine learning textbook.

[12]"Data Clustering: Algorithms and Applications," a comprehensive book on clustering methods and practices.