Hierarchical Clustering and Automatic Labelling of Customer Reviews

GROUP-2
Harshith Reddy Thammineni (ED18B036)
Nikitha S R (BE18B011)
Swathi V (ED18B034)
S. Kishan Rao (ED18B033)
Mudita Toshniwal (AE18B107)
Krisha Ketan Shah (ED18B017)

***Abstract*—Text clustering is an unsupervised learning task that can be used to classify text data (such as reviews in this case) into various clusters based on their similarity. In this article, we explore a two-level hierarchical clustering algorithm to cluster review data obtained from an online platform. Metrics to evaluate the clustering obtained, as well as methods to enhance the model performance are also discussed.**

I. Introduction

The amount of text data being generated in the recent years has exploded exponentially. It is essential for organizations to have a structure in place to mine insights from the text being generated. From social media analytics to grouping reviews on online platforms, dealing with text data has never been more relevant.

Text clustering is an **unsupervised** machine learning task, that groups texts into various clusters in such a way that the text data in the same cluster are more similar than those in different clusters. This is different from classification in the sense that classification deals with labelled data (supervised), whereas clustering deals with unlabelled data (unsupervised). Text clustering is done by first representing the text data as a numerical feature vector, and then applying clustering algorithms such as k-means to split them into groups.

In this article, we apply a **two level hierarchical k-means clustering** to text data obtained from **reviews**. We also explore a method to label the clusters automatically to improve readability of the clusters.

The flow of this article is as follows. The sections on *text mining* and *clustering* give a broad overview of the topics, not in the context of the given problem statement. We then look at the *data*, wherein we also mention approached followed for data cleaning. We then describe our *approach* to solve the given problem, followed by the *results* and *conclusions*.

II. Text Mining

Text mining is the conversion of unstructured text data into structured data (vectors) that can be used in machine learning algorithms. The steps involved in text mining are:

A. Text Preprocessing

The raw text obtained can be noisy, and information can be hidden in stopwords and varied forms of the same words. In order to process the text data, we perform the following steps:

1) Removing stopwords: Words such as 'and', 'the', etc., which do not add any meaning to the sentence, but have very high frequency, are called stopwords. These words are removed to retain only the terms which are more meaningful to get an accurate representation in the feature vector.

2) Transforming: The words are all tranformed to lower case, and punctuations are removed.

3) Normalization: Text normalization is the process of transforming a text into a canonical (root) form. Stemming (removing the last few letters of the word) and lemmatization (deriving the root form of the word) techniques are used for deriving the root word.

B. Feature Extraction

Feature extraction is the conversion of text data to feature vectors that can be used in machine learning models. The following techniques can be used to convert text to vectors:

1) Bag-of-Words: To find the bag of words representation of a given sentence, we first create a vocabulary of all possible words or n-grams. The representation of each sentence is then the count of each word that is present in the sentence. The intuition behind this representation is that similar sentences contain similar words.

2) TF-IDF: The approach of TF-IDF is the rescale the words by their term frequency (how many times the appear in the sentence), and by their inverse document frequency (inverse of the number of times the word appears across all sentence, which measures the informativeness of the word). Thus, each word score (from the bag of words vectors) are scaled by the TF*IDF, which gives us the final word encoding.

The major drawback of this approach is that as the size of the vocabulary increases, the size of the encoding increases proportionally.

3) Word embeddings: Word representations can also be learnt using word embeddings, the most common ones of which are *Word2Vec* and *GLoVe* representations.

Once the representations are learnt, these can be used in clustering algorithms explained in the next section.

III. Clustering

Clustering is the task of dividing data points such that the points in the same cluster are more *similar* than ones in different clusters. The similarity between data points is measured using the distance between them, either by the Euclidean distance, or some other similarity criteria such as the cosine similarity. There are numerous clustering algorithms, of which we will look at *hierarchical clustering* and *k-means* clustering in this paper.

A. K-means clustering

K-means clustering works by first defining the target number of clusters (k), and assigning k centroids randomly, one for each cluster. Every data point is now allocated to one of the clusters based on their distance from the centroids. The centroids are then recalculated as the mean of the data points in each cluster. This is performed iteratively until the centroids converge to the true mean of the clusters.

The major limitation of this algorithm is that the number of clusters has to be declared initially, and cannot be learnt from the data dynamically.

B. Hierarchical clustering

Hierarchical clustering is an algorithm used to split data into clusters at various clusters. For example, the entire dataset can be divided into 3 top level clusters, each of which can be further divided into 5 clusters at the next level. Such clustering algorithms are especially useful in text analysis. There are two forms of hierarchical clustering:

1) Agglomerative Clustering: Agglomerative clustering works by first considering each data point it's own cluster, and then successively merging clusters which are the closest to each other. This iterative process continues until all the clusters are merged together.

2) Divisive Clustering: Divisive clustering is a top-down approach, where initially all data points are considered to be part of the same cluster, and are recursively divided moving down the hierarchy.

C. Hierarchical Clustering Using k-Means

A hierarchical k-means clustering approach has been adapted from [1], wherein we first cluster the data points into n_1 clusters, which are each divided into n_2 clusters each. This approach is explained in Fig. 1.

D. Evaluation of clustering

When the ground truth is not known, *silhouette score* is a common metric used to understand the quality of the clusters formed. The silhouette coefficient is calculated using the mean intra-cluster distance a and the mean nearest-cluster distance b for each sample. The silhouette coefficient for a sample is $(b - a)/max(b, a)$.

The range of the silhouette coefficient is (-1, 1). Values near 0 indicate overlapping clusters. Negative values generally indicate that a sample has been assigned to the wrong cluster, as a different cluster is more similar.

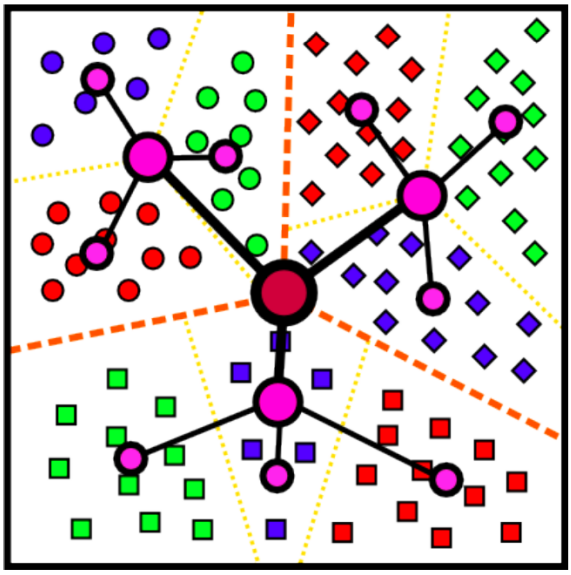

Fig. 1. Hierarchical k-means clustering (Source: [1])

IV. Data

The given dataset contains customer reviews obtained for a product, obtained from an online shopping portal. Our task is to divise a 2-level clsutering algorithm to cluster the reviews, and to automatically label them based on the words present in each cluster.

The fields present in the given dataset are: *date, Before Translation,* ***Review****, Author, Location, Before Pronoun Resolution, Brand, Number of Ratings, Price, Product Rating, Review Count, Review Rating, Review Title, Root, SKU, Source,* ***Aspect****,* ***Context Aspect****, Sentence, Combined Max Score, Overall Sentiment, Sentiment Score, Sentiment Confidence, Clustering Confidence, Context Aspect Group, Aspect Sentiment, Aspect Sentiment Score, Context Aspect Sentiment, Context Aspect Sentiment Score, Context Aspect Group Sentiment, Context Aspect Group Sentiment Score*. Since the dataset contained many fields that we did not require, only the fields mentioned in bold were retained. The *aspect* field was used to obtain some kind of ground truth for the level 1 clustering, and the *context aspect* for level 2 clustering.

A. Data Cleaning

To clean (preprocess) the data, the following steps were performed:

- Tokenize the data
- Removing stopwords
- Lemmatize (to get the root form of the words)

V. Approach

The various steps followed to hierarchically cluster the given text data were:

- Feature encoding
- (Level-1) Clustering using k-means
- Automatic labelling
- Second level clustering and labelling for data points in each level-1 cluster

A. Feature Encoding

TF-IDF was used to convert each review (sentence) to a feature vector, containing **200** features. This is a hyperparameter than can be tuned.

B. Level-1 Clustering Using k-Means

The number of clusters was set to 12 for the first level k-means clustering. This was tuned by plotting the *average distortion* for various values of k and choosing the best value of k. This is called the elbow method, and is shown in Fig. 2.

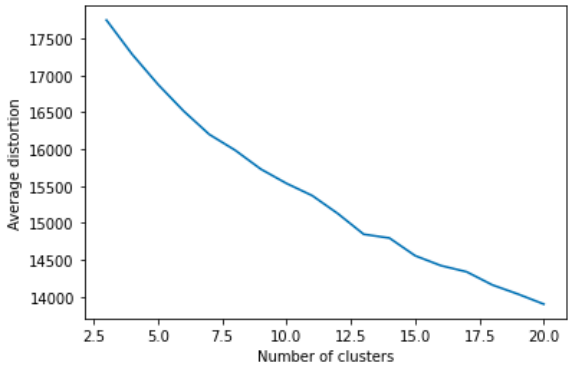

Fig. 2. Elbow plot to choose ideal number of clusters

The k-means algorithm was slightly modified such that any two clusters with a mean distance less than 0.2 would be merged, even if it meant a reduction in the number of clusters.

C. Automatic Labelling of Clusters

To automatically label the clusters, the following approach was followed:

- The preprocessed text was used to extract a list of all unique words and bigrams in the reviews.
- The frequency of words and bigrams over all the reviews in that particular cluster was calculated, and the one with the highest frequency was chosen as the text label.

D. Second Level Clustering and Labelling

For each level-1 cluster, the data points were further classified into $n_2 = 3$ clusters. Again, clusters which were less than 0.2 units apart or had the same label were merged into a single cluster, even if if meant a reduction in the number of clusters.

For labelling of level-2 clusters, if the label produced by the automatic labelling algorithm was same as the name of the level-1 cluster name, then the second most frequent bigram was taken as the cluster label.

E. Hyperparameters than can be tuned

The hyperparameters than can be tuned in the above algorithm are:

- Number of features (during feature encoding using TF-IDF)
- n_1 Number of level 1 clusters
- n_2 Number of level 2 clusters

F. Algorithm

The algorithm described above is summarized in Fig. 3.

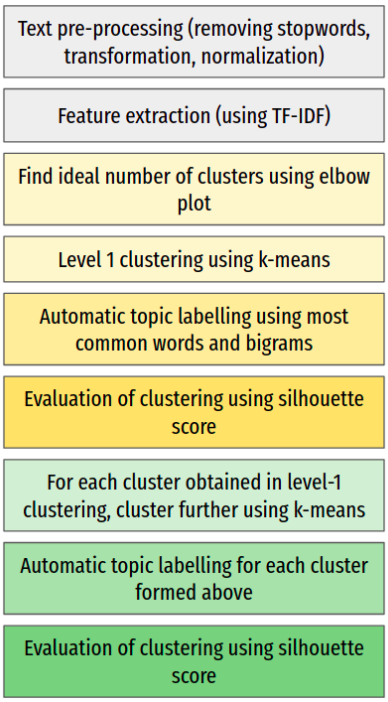

Fig. 3. Summary of the hierarchical k-means clustering and automatic labelling algorithm

VI. Results

A. Level 1 Clustering

The results of the clustering algorithm are compared to the ground truth using visualization. The given (ground truth) clustering is shown in Fig. 4. We observe that the clusters are not clearly differentiable in the 3 dimensional plot.

The clustering produced (including the labels), and the silhouette score are shown in Fig. 5. We see that though the silhouette score is low, the clusters are visually similar to the ones in Fig. 4.

B. Level 2 Clustering

The clusters labels formed in the level 2 clustering (along with their silhouette scores) are shown in Fig. 6. We see that most of the clusters formed (and labels henceforth) are meaningful. Each level-1 cluster and the subsequent clusters formed are shown in Fig. 7 to Fig. 18.

The silhouette scores for all the level 1 and level 2 clusterings are not very high, however, the plots show that the

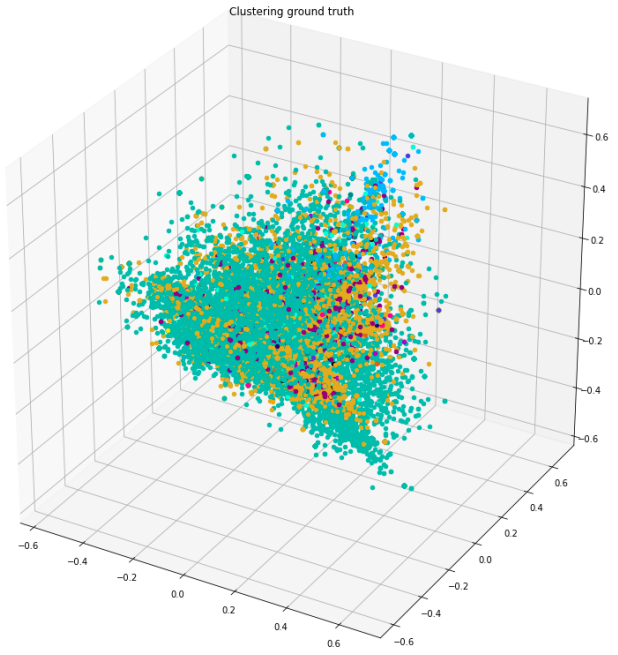

Fig. 4. Ground truth for level 1 clustering (without labels)

```
muscle
        sore muscle, muscle pain, pain
        Silhouette score =  0.28554318079245633
cramp
        night, theraworx, work
        Silhouette score =  0.09039447773248474
patch
        salonpas, pain
        Silhouette score =  0.06423449591123355
help
        help pain, pain
        Silhouette score =  0.3591247202786995
pain
        work, knee, like
        Silhouette score =  0.011232215614377037
work great
        great
        Silhouette score =  0.278030624369103
good
        good price, work, good product
        Silhouette score =  0.13328458001208585
work
        work better, pain, product work
        Silhouette score =  0.0039505859380714456
ache pain
        ache
        Silhouette score =  0.19900377194578575
pain relief
        relief
        Silhouette score =  0.18382886941473253
product
        best, great product, pain
        Silhouette score =  0.2132118162806623
review collected
        collected promotion
        Silhouette score =  0.12069495761536962
```

Fig. 6. The label tree produced by the 2-level hierarchical clustering algorithm

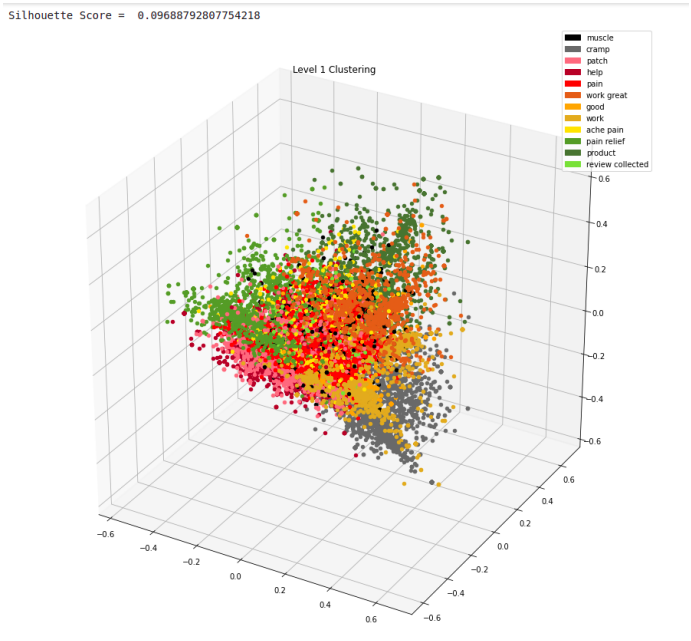

Fig. 5. Level 1 clustering using k-means

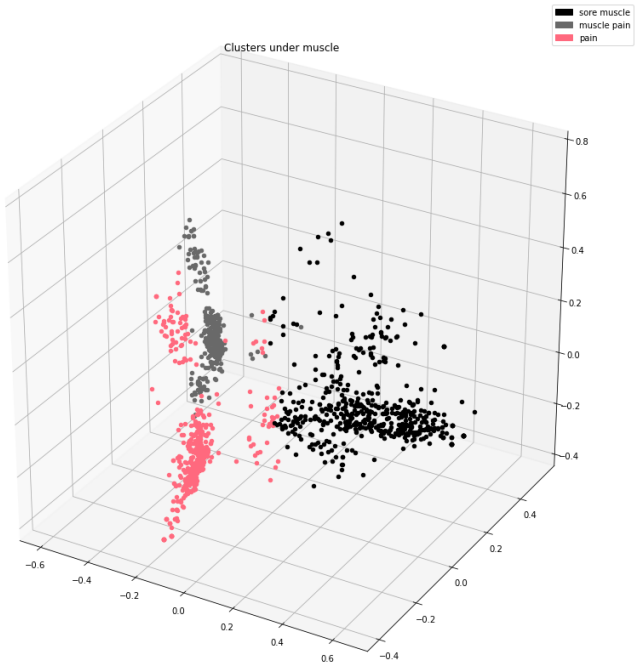

Fig. 7. Second level clustering of the *muscle* cluster

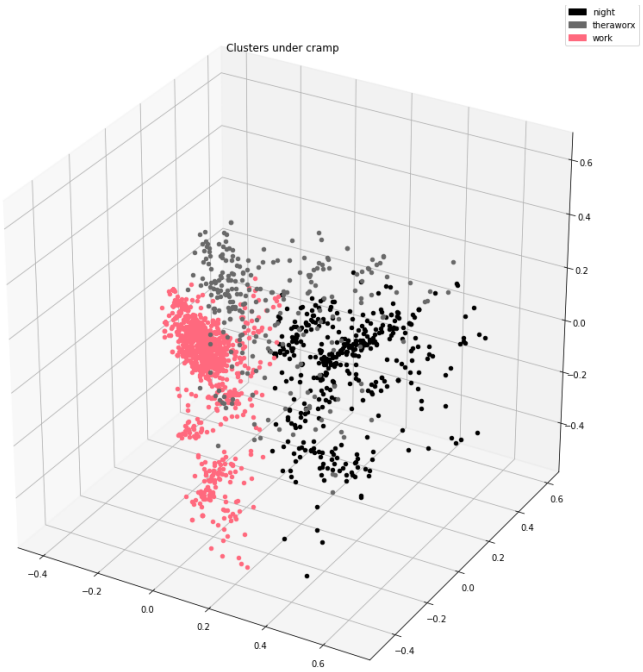

Fig. 8. Second level clustering of the *cramp* cluster

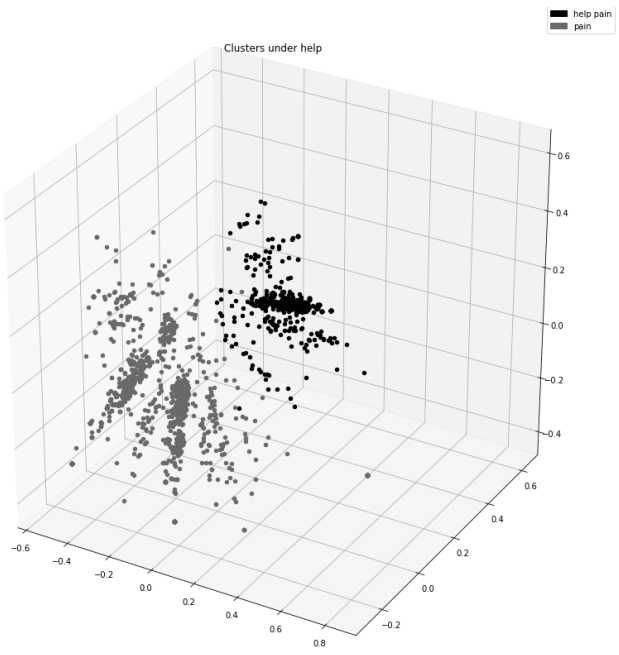

Fig. 10. Second level clustering of the *help* cluster

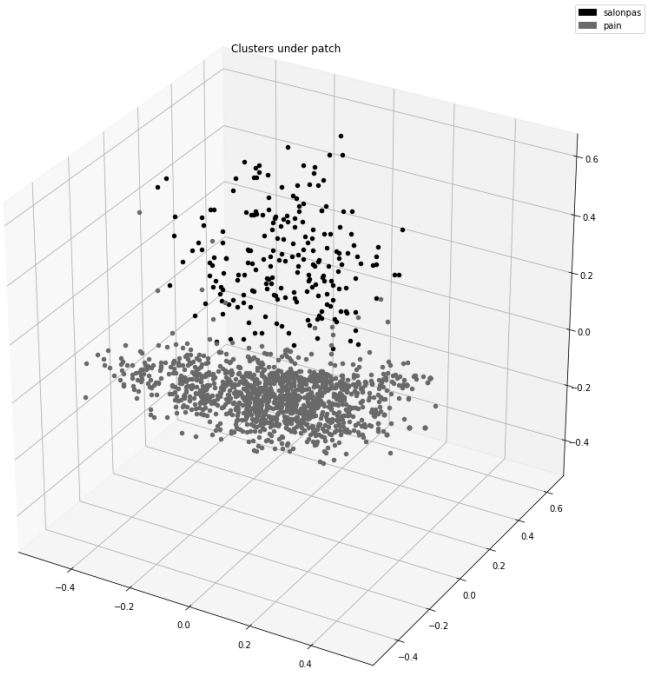

Fig. 9. Second level clustering of the *patch* cluster

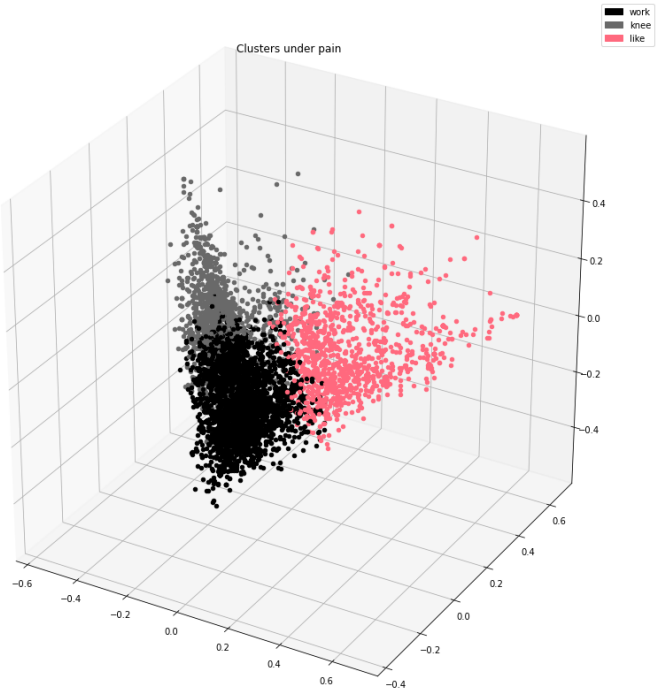

Fig. 11. Second level clustering of the *pain* cluster

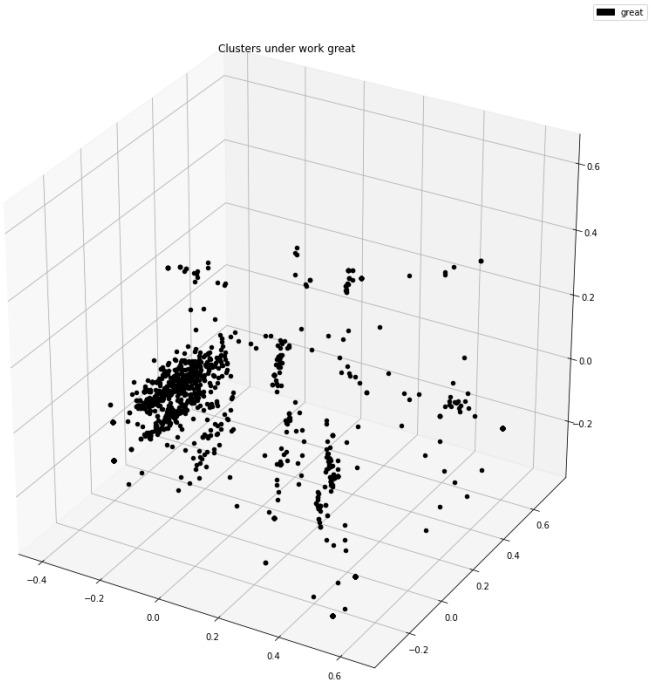

Fig. 12. Second level clustering of the *work great* cluster

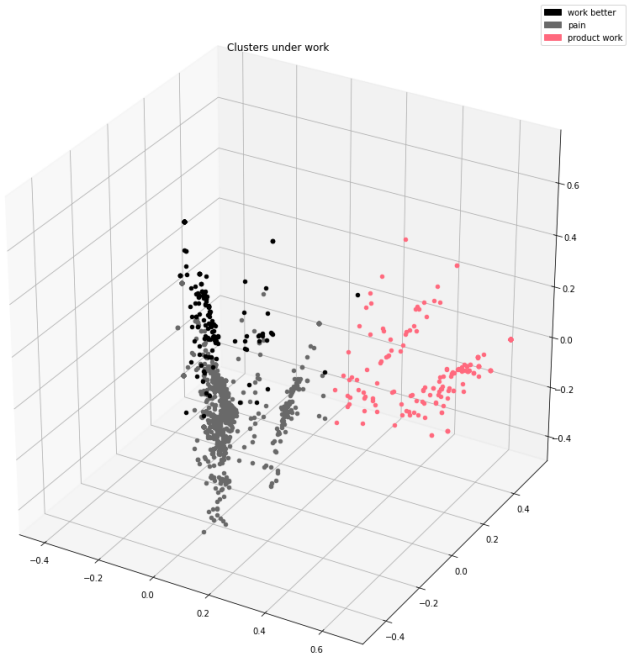

Fig. 14. Second level clustering of the *work* cluster

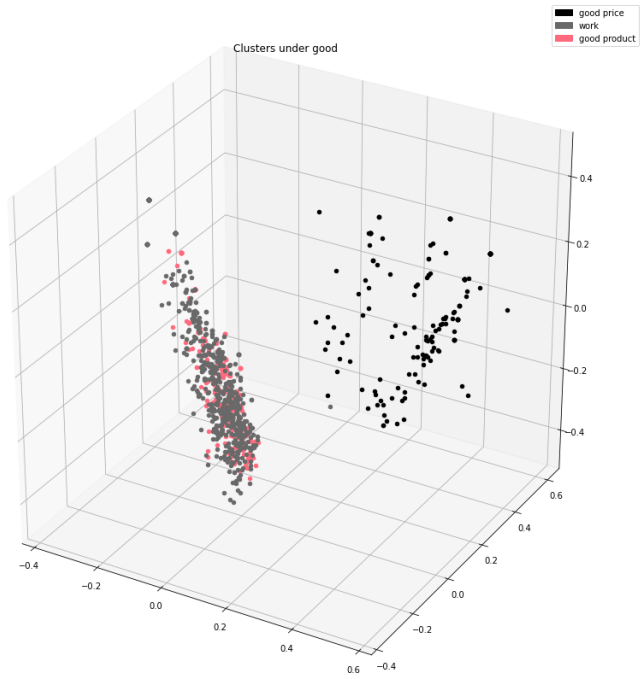

Fig. 13. Second level clustering of the *good* cluster

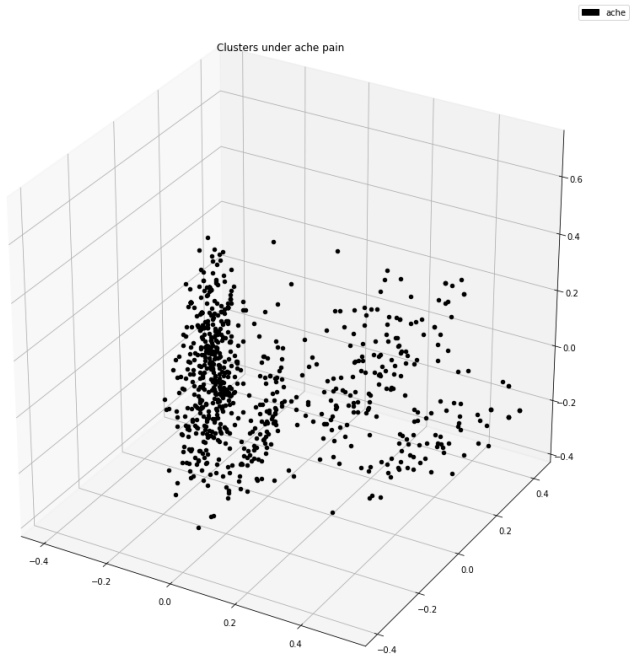

Fig. 15. Second level clustering of the *ache pain* cluster

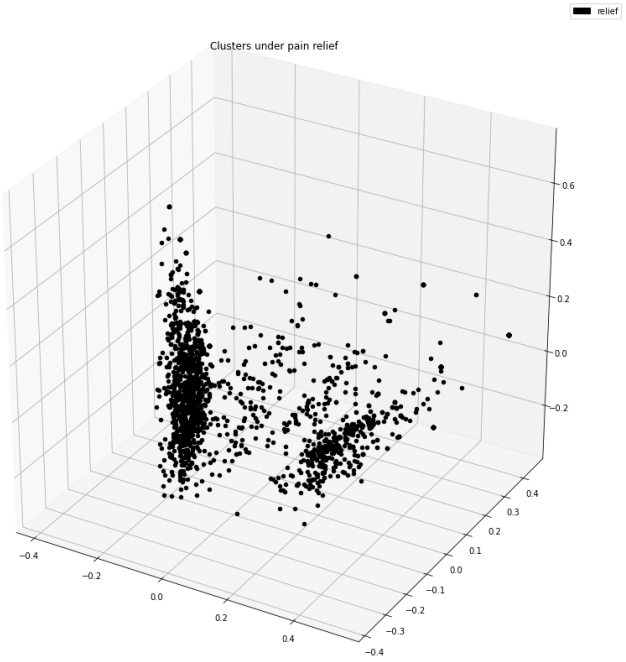

Fig. 16. Second level clustering of the *pain relief* cluster

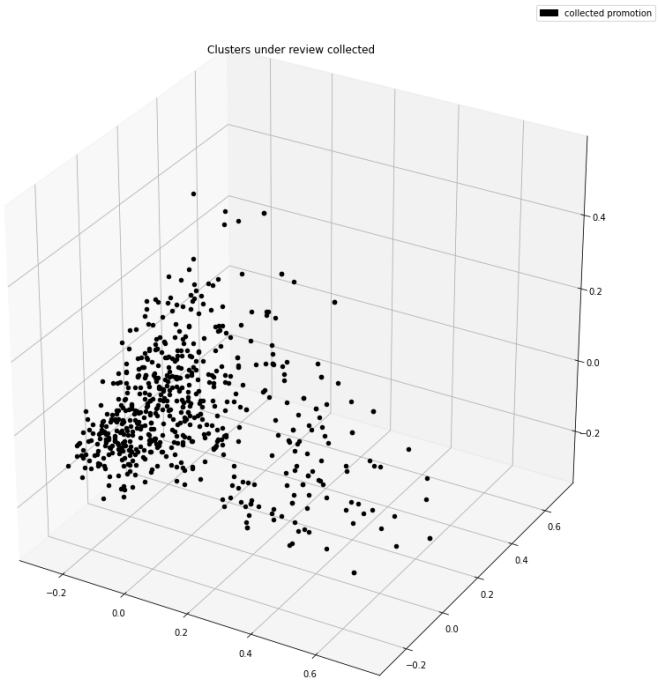

Fig. 18. Second level clustering of the *review collected* cluster

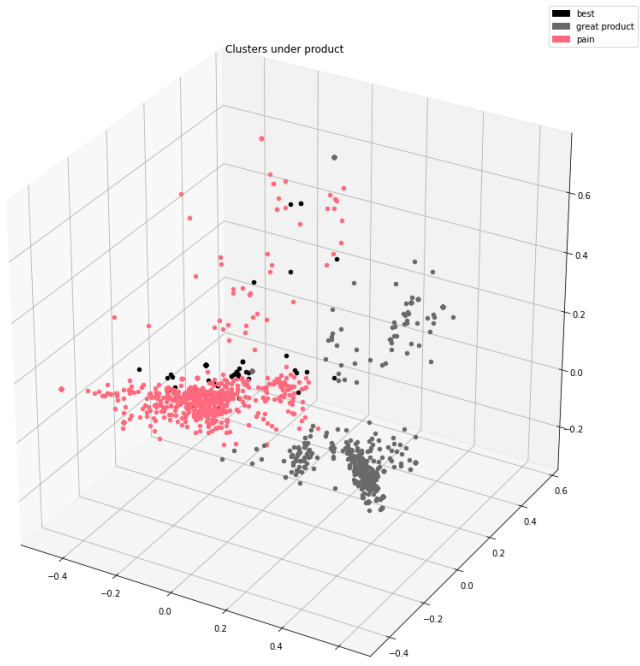

Fig. 17. Second level clustering of the *product* cluster

clusters are well formed, which is also reflected in the labels. The algorithm can be further tuned by changing the number of features in the TF-IDF vectorization, as well as the number of clusters n_1 and n_2.

VII. Conclusions

The above clustering algorithm performs well in clustering the given dataset containing reviews. The hyperparameters can be tuned such that the same algorithm can be applied to any text dataset containing sentences. For more complicated datasets, other text embedding techniques can also be used to improve the performance of the model.

References

[1] Harry Gifford, *Hierarchical k-Means for Unsupervised Learning*, Carnegie Mellon University.