Lab8

1. Team Details

Name	USC ID
Chenxiao Yu	6024079123
Yiqing Hong	4395913002

2. Github Link:
https://github.com/AiChiMoCha/SP25_DSCI560/tree/main/lab8

3. YouTube Link
https://youtu.be/ynUCKuDckEY

4. Accuracy Comparison Doc2Vec Embeddings
In Part1.py, we use THREE different doc2vec configurations (vector_size, min_count, epochs), especially changing vector size

```
# Define three different Doc2Vec configurations
doc2vec_configs = [
    {"vector_size": 50, "min_count": 2, "epochs": 40},
    {"vector_size": 100, "min_count": 2, "epochs": 40},
    {"vector_size": 200, "min_count": 2, "epochs": 40}
]
```

Then we convert text to lowercase, remove special characters, split into tokens, and filter stopwords. We process each configuration.

```
for config in doc2vec_configs:
    print(f"Running Doc2Vec with vector_size = {config['vector_size']}")

    # Initialize and train the Doc2Vec model
    model = Doc2Vec(vector_size=config["vector_size"], min_count=config["min_co
    model.build_vocab(tagged_docs)
    model.train(tagged_docs, total_examples=model.corpus_count, epochs=model.ep

    # Infer vectors for each document
    doc_vectors = []
    doc_ids = []
    for doc in tagged_docs:
        vec = model.infer_vector(doc.words)
        doc_vectors.append(vec)
        doc_ids.append(doc.tags[0])

    doc_vectors = np.array(doc_vectors)
    # Normalize vectors (L2 normalization) for cosine distance comparison
    norm_vectors = normalize(doc_vectors)

    # Cluster the normalized vectors using KMeans
    kmeans = KMeans(n_clusters=n_clusters, random_state=42)
    clusters = kmeans.fit_predict(norm_vectors)

    # Build a DataFrame to store the clustering results
    results_df = pd.DataFrame({
        "id": doc_ids,
        "title": posts_df.set_index('id').loc[doc_ids, 'title'].values,
        "cluster": clusters,
        "vector_size": config["vector_size"]
```

Finally, we evaluate which configuration is better. We use 5 different methods to show their performance.
Quantitative:

a. Silhouette Score
   • Measures the compactness and separation of clusters, with a value range of [-1, 1].
   • Calculation formula:

$$S = \frac{b - a}{\max(a, b)}$$

   Where:
   • a is the average distance from a point to other points in the same cluster (intra-cluster distance).
   • b is the average distance from the point to the nearest other cluster (inter-cluster distance).
   The closer it is to 1, the better the clustering effect is.

b. Davies-Bouldin Index (DBI)
   • Measures the similarity between clusters, the lower the value, the better the clustering effect.
   • Calculation formula:

$$DBI = \frac{1}{n} \sum_{i=1}^{n} \max_{i \neq j} \frac{s_i + s_j}{d_{ij}}$$

   Where:
   • s_i and s_j are the divergences of the two clusters (the average distance of the data within the cluster).
   • d_ij is the distance between the centers of the two clusters.

c. Within-Cluster SSE
   • Calculates the sum of the squared distances from each point to the center of its cluster to measure data compactness.
   • Lower values indicate tighter data within the cluster

Qualitative:

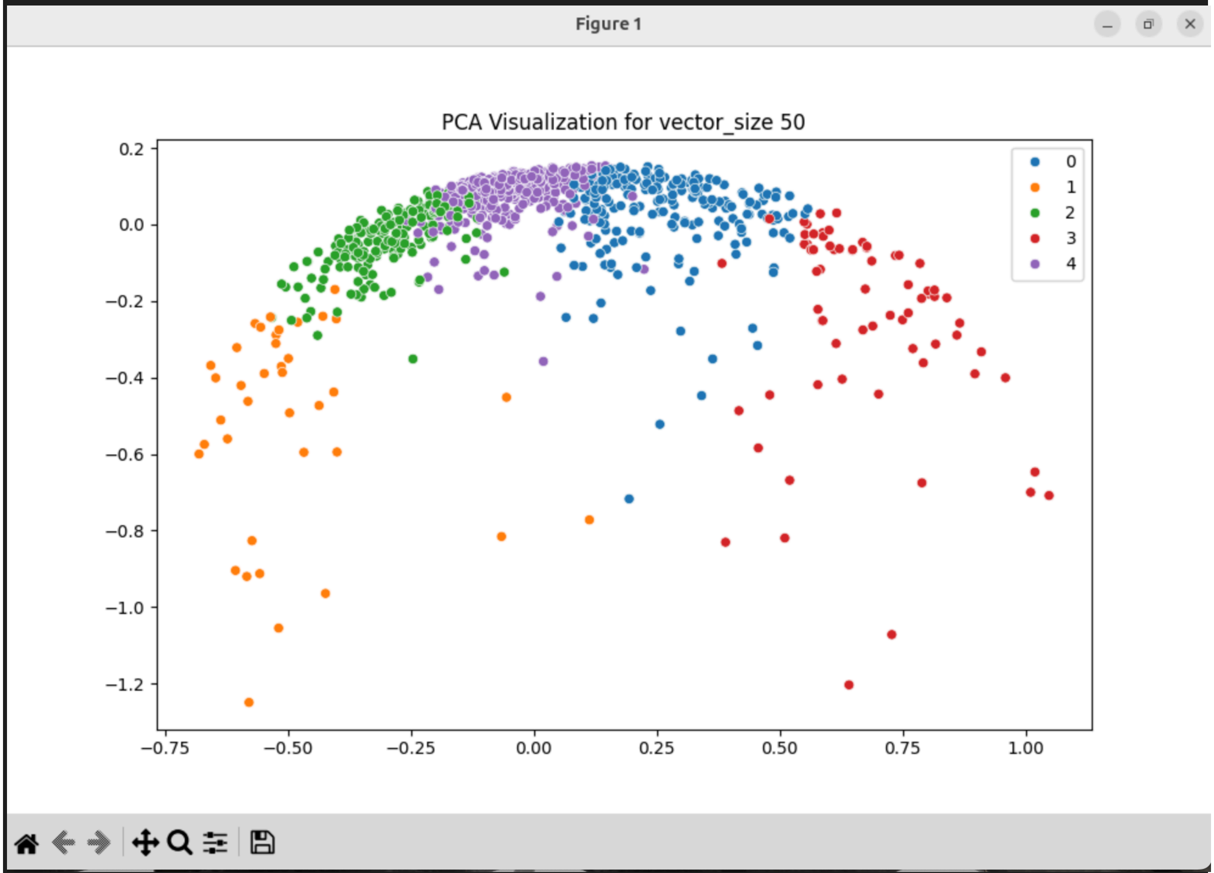
d. Low-dimensional visualization (PCA)
   • Since the vectors generated by Doc2Vec are of high dimension (50, 100, 200), PCA can be used to reduce the dimension to 2D for visualization to see the distribution of different clusters.
   • If the clustering boundaries generated by Doc2Vec of a certain vector size are clearer, it means that it works better

e. Intra-cluster keyword analysis
   • Check the high TF-IDF words in each cluster to see if they are semantically reasonable.
   • If a certain Doc2Vec configuration makes posts in the same cluster more semantically consistent, it means it performs better.

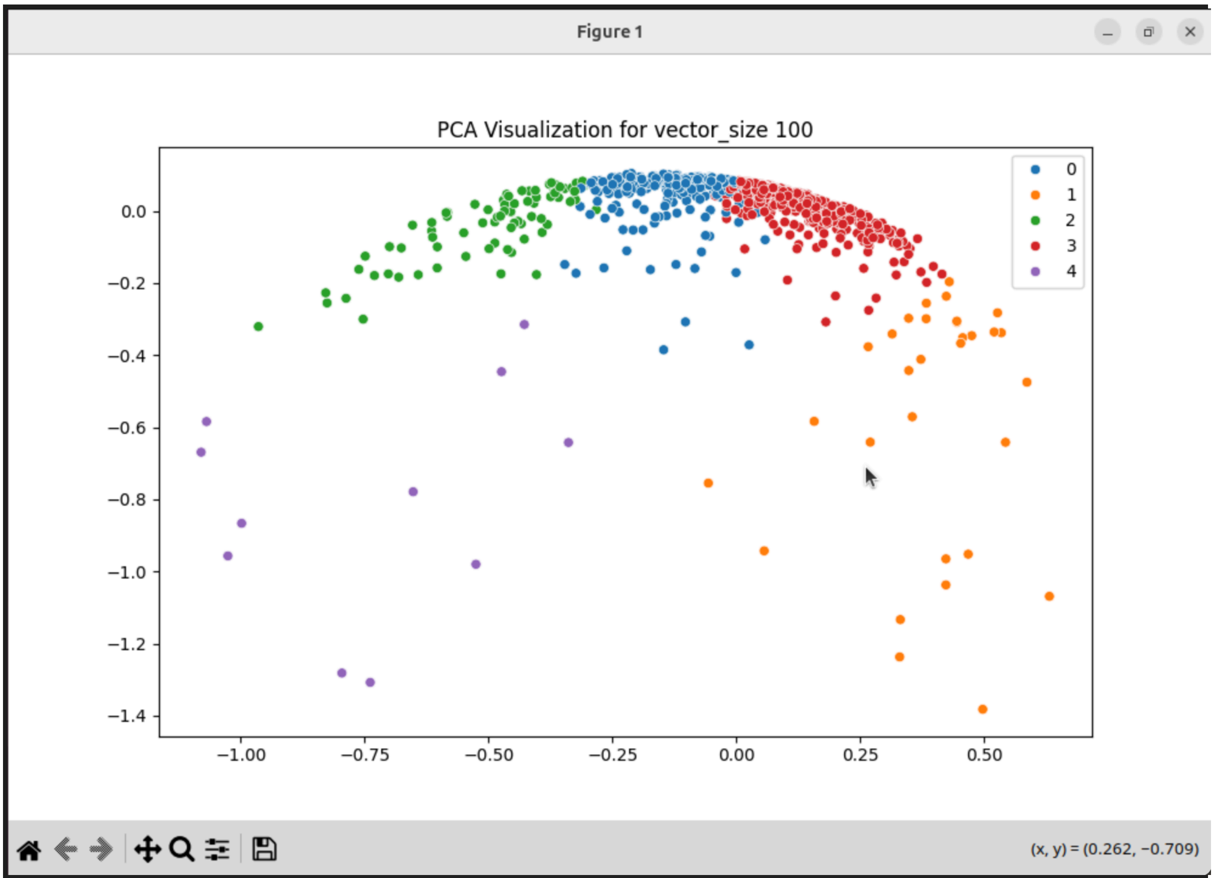
Results:

```
Running Doc2Vec with vector_size = 50


Silhouette Score for vector_size 50: 0.22758077085018158
Davies-Bouldin Index for vector_size 50: 1.854970606485179
Within-Cluster SSE for vector_size 50: 134.7251434326172
Cluster 0:
['solar', 'battery', 'world', 'new', 'power', 'energy', 'water', 'fusion', 'ev',
'batteries']
Cluster 1:
['human', 'new', 'ai', 'brain', 'robot', 'like', 'mit', 'cells', 'robots', 'blood']
Cluster 2:
['new', 'brain', 'cancer', 'researchers', 'robot', 'human', 'cells', 'ai',
'disease', 'treatment']
Cluster 3:
['energy', 'water', 'heat', 'battery', 'fuel', 'co2', 'new', 'world', 'power',
'scientists']
```

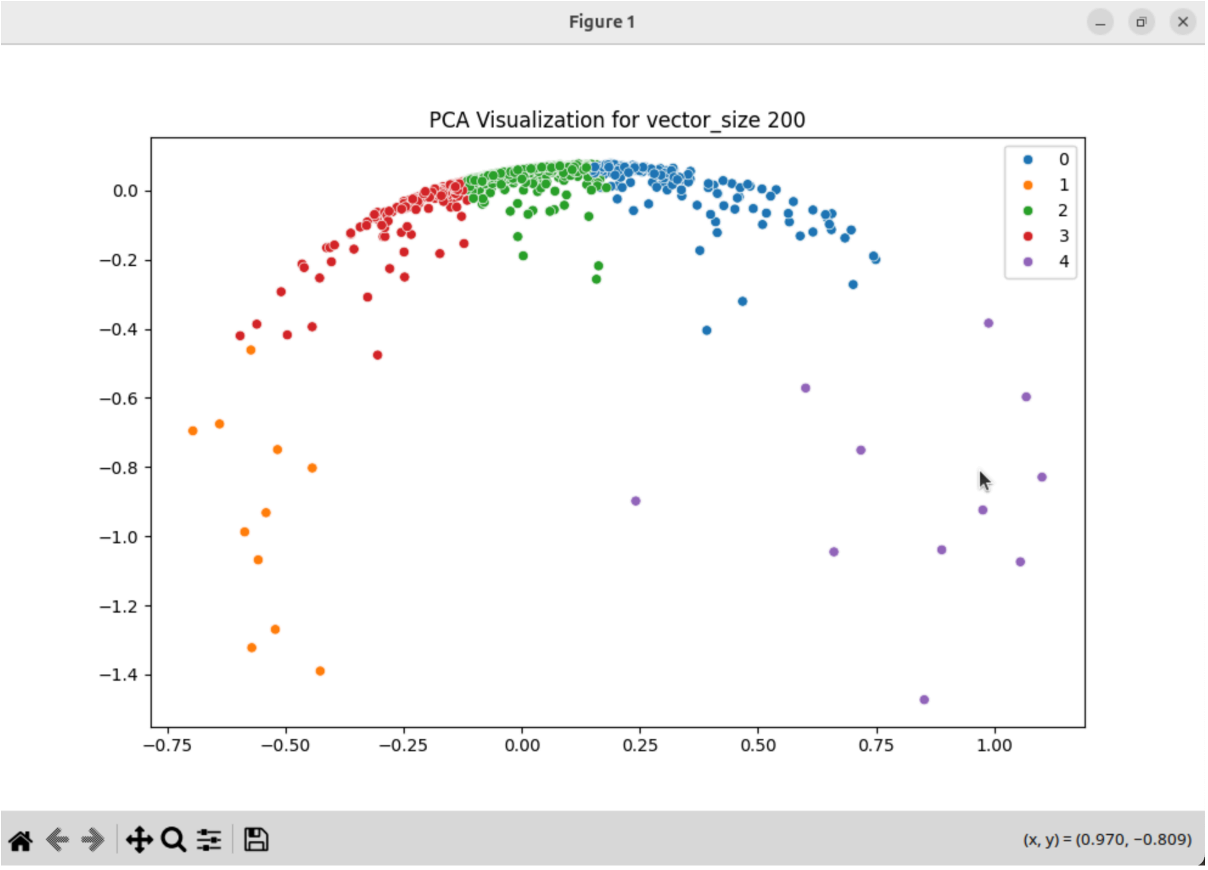
```
Cluster 4:
['new', 'scientists', 'quantum', 'ai', 'robot', 'researchers', 'water', 'light',
'tech', 'world']
```




```
Running Doc2Vec with vector_size = 100

Silhouette Score for vector_size 100: 0.28530460596084595
Davies-Bouldin Index for vector_size 100: 1.650603571127905
Within-Cluster SSE for vector_size 100: 94.66605377197266
Cluster 0:
['world', 'new', 'power', 'solar', 'water', 'battery', 'hydrogen', 'based',
'batteries', '1st']
Cluster 1:
['human', 'robot', 'ai', 'new', 'brain', 'humanoid', 'blood', 'like', 'robots',
'mit']
Cluster 2:
['energy', 'battery', 'water', 'power', 'heat', 'batteries', 'fuel', 'new', 'ev',
'000']
Cluster 3:
['new', 'ai', 'researchers', 'robot', 'quantum', 'brain', 'scientists', 'human',
'cancer', 'robots']
Cluster 4:
['nuclear', 'fusion', 'new', 'battery', 'boost', 'reactor', 'efficiency', 'space',
'tech', 'water']
```




```
Running Doc2Vec with vector_size = 200

Silhouette Score for vector_size 200: 0.3837256133556366
Davies-Bouldin Index for vector_size 200: 1.3248065991697935
Within-Cluster SSE for vector_size 200: 58.91192626953125
Cluster 0:
['energy', 'battery', 'water', 'batteries', 'heat', 'fuel', 'power', 'new',
'world', 'solar']
Cluster 1:
['robot', 'brain', 'new', 'like', 'test', 'helps', 'robots', 'muscles',
'technique', 'hand']
Cluster 2:
['new', 'world', 'ai', 'scientists', 'quantum', 'solar', 'researchers', 'robot',
'water', 'powered']
Cluster 3:
['brain', 'new', 'human', 'researchers', 'ai', 'cancer', 'robot', 'cells',
'robots', 'heart']
Cluster 4:
['nuclear', 'battery', 'water', 'new', 'efficiency', 'fusion', 'storage',
'reactor', 'heat', 'energy']
```



Through the quantitative methods, we can see that vector_size = 200 has the largest Silhouette Score, smallest DBI and within-cluster SSE. The clustering boundaries of vector_size = 200 are also clearer. So, it is the best configuration among these three.

5. Embeddings based on Word2Vec and Bag-of-Words
In Part2.py, we build the corpus and collect document metadata, and train the Word2Vec model on the entire corpus. We define three different bin configurations as before.

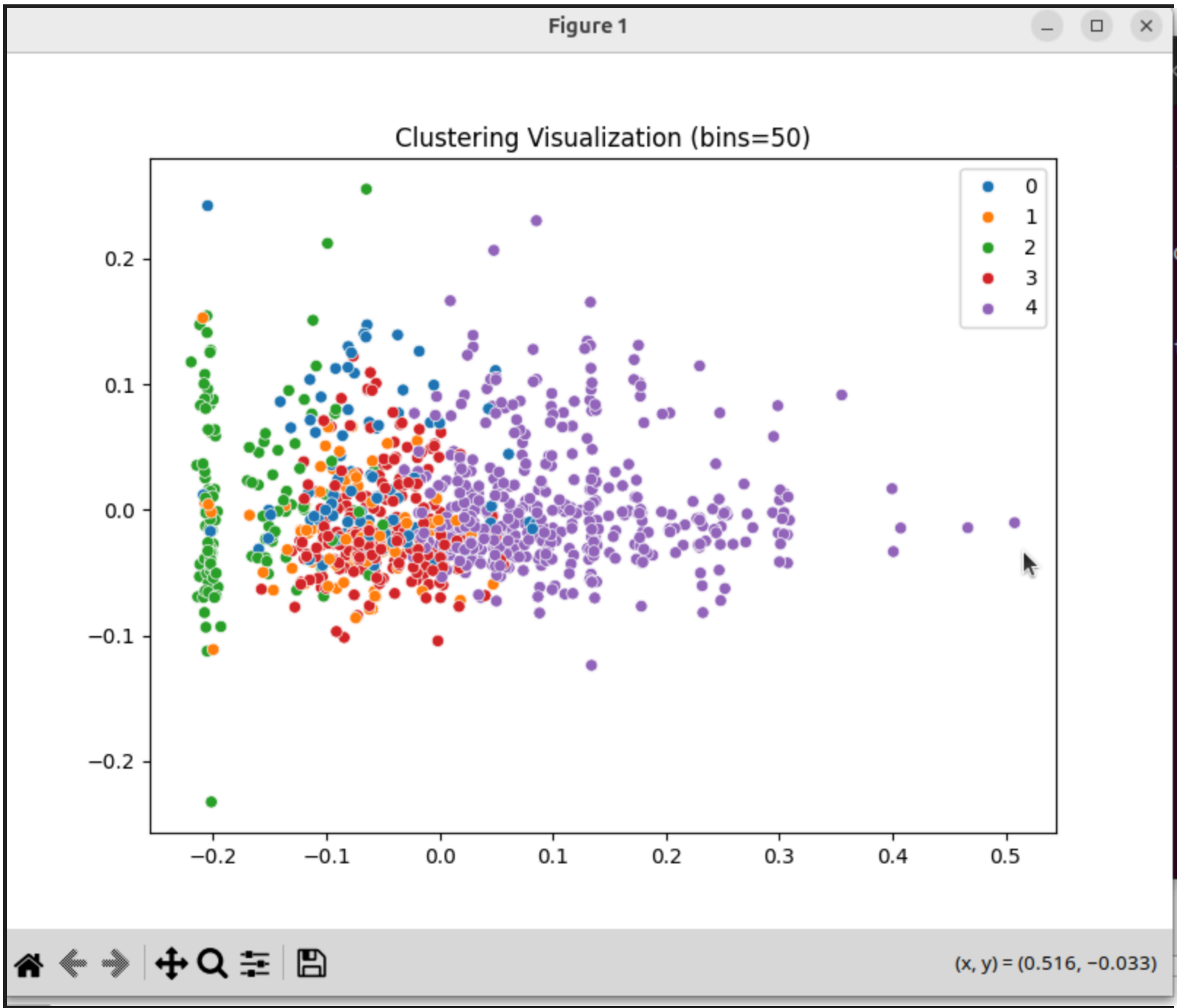
```
# Define three different bin configurations (dimensions)
bin_configs = [50, 100, 200]
```

Then we do the k-means clustering.

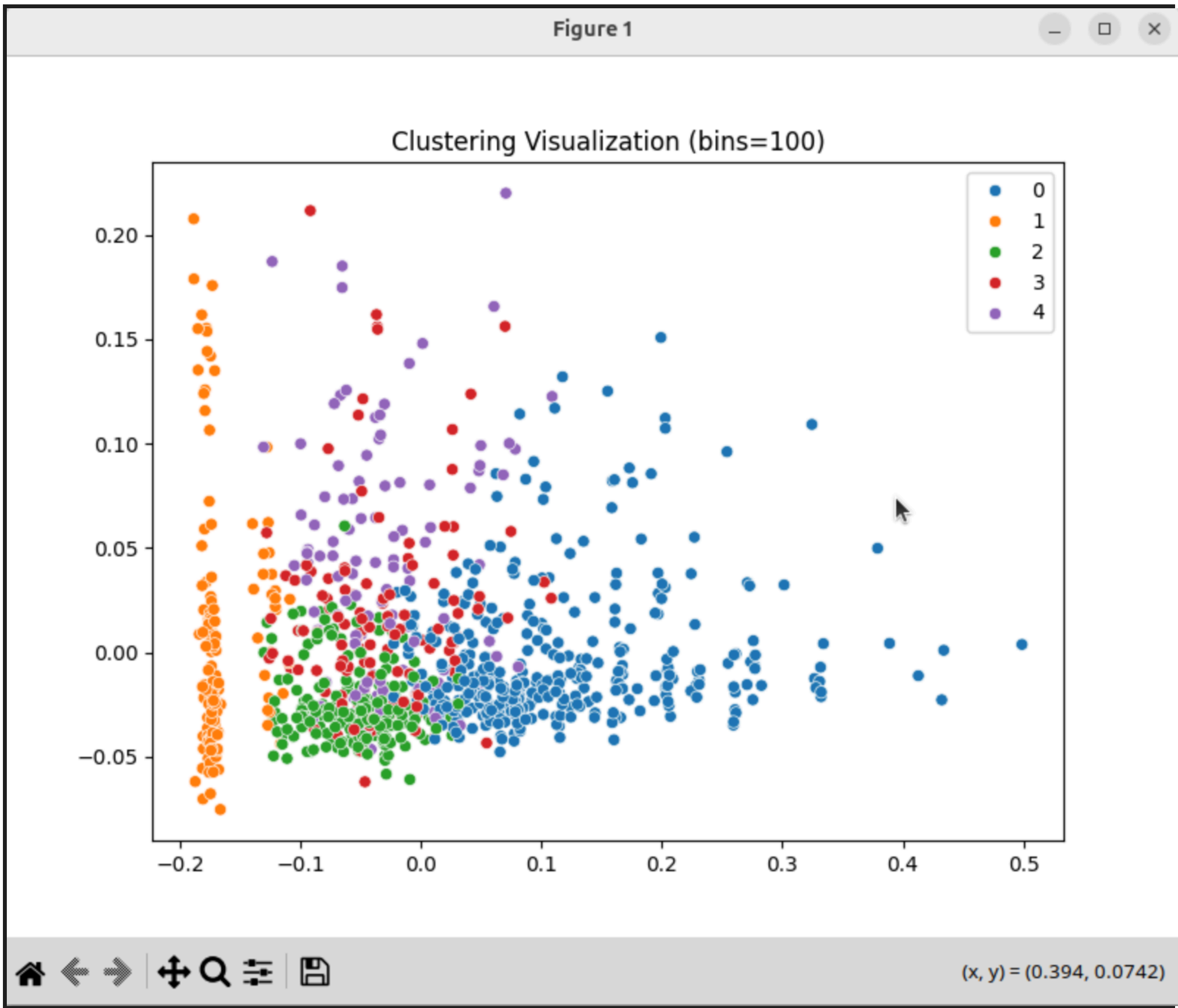
```python
for bins in bin_configs:
    print(f"\nRunning Word2Vec-Bag-of-Words with {bins} bins")

    # Extract all words and their vectors from the model
    vocab_words = list(word2vec_model.wv.key_to_index.keys())
    word_vectors = np.array([word2vec_model.wv[word] for word in vocab_

    # Cluster the word vectors into 'bins' clusters using KMeans
    kmeans_words = KMeans(n_clusters=bins, random_state=42)
    word_cluster_labels = kmeans_words.fit_predict(word_vectors)

    # Create a dictionary mapping each word to its bin/cluster label
    word_to_cluster = {word: label for word, label in zip(vocab_words,

    # Generate a normalized histogram vector for each document
    doc_vectors = []
    for tokens in corpus:
        hist = np.zeros(bins)
        word_count = 0
        for token in tokens:
            if token in word_to_cluster:
                hist[word_to_cluster[token]] += 1
                word_count += 1
        if word_count > 0:
            hist = hist / word_count  # Normalize the histogram
        doc_vectors.append(hist)

    doc_vectors = np.array(doc_vectors)

    # Cluster the document vectors using KMeans (normalize for cosine s
```

Finally, we evaluate these configurations using the same methods as above.

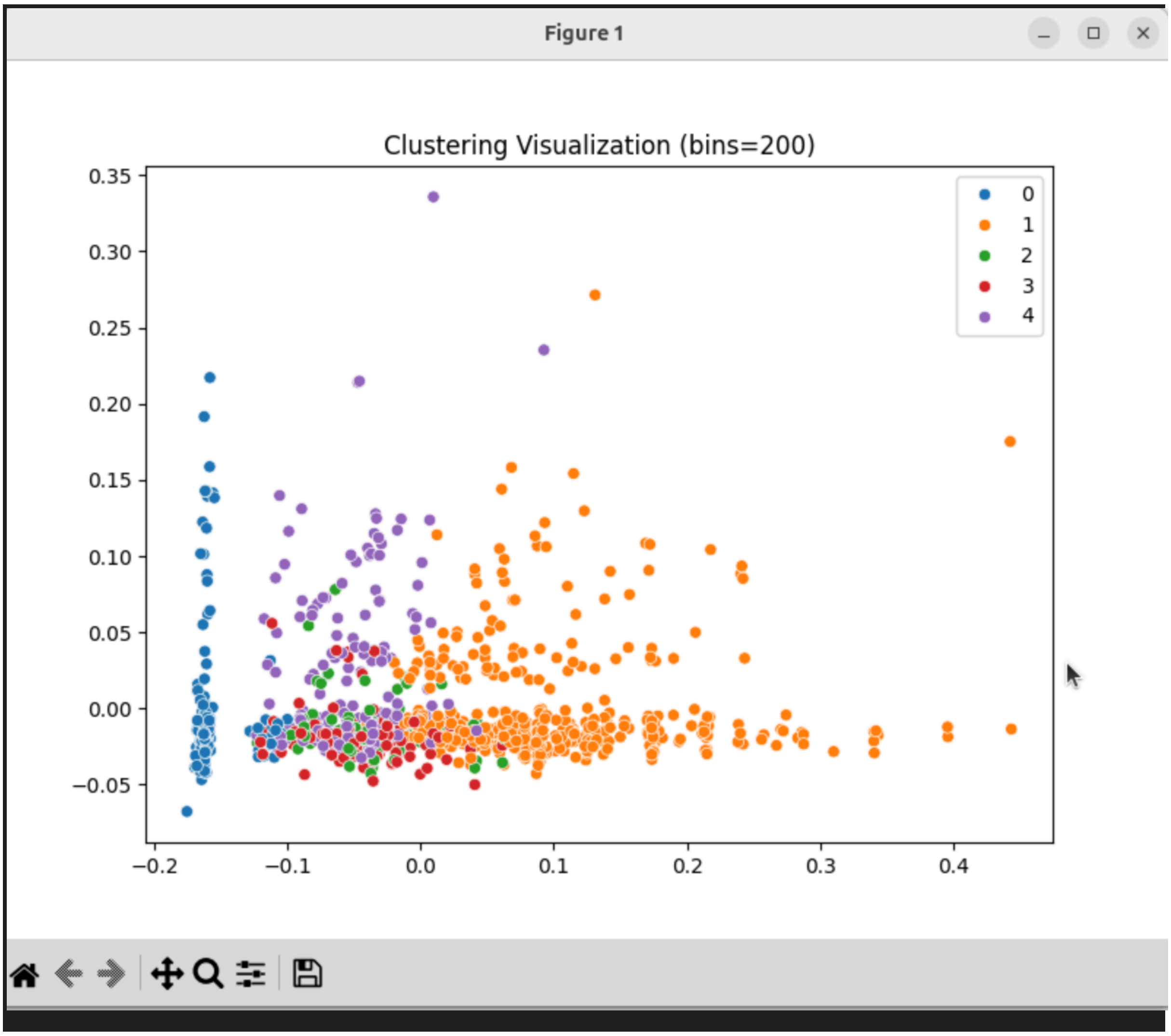
```
Running Word2Vec-Bag-of-Words with 50 bins


Silhouette Score (bins=50): 0.05002492601571487
Davies-Bouldin Index (bins=50): 4.261869022660962
Within-Cluster SSE for bins=50: 521.9404641439686
Cluster 0:
['reactor', 'co2', 'new', 'world', 'plastic', 'scientists', 'treatment', 'field',
'patients', '1st']
Cluster 1:
['ai', 'new', 'neutrons', 'fusion', 'world', 'years', 'researchers', 'wetsuits',
'air', 'high']
Cluster 2:
['powered', 'muscle', 'robot', '3d', 'world', 'lens', 'satellites', 'brew',
'connect', 'new']
Cluster 3:
['new', 'researchers', 'scientists', 'world', 'ai', 'self', 'robot', 'blood', '3d',
'technology']
Cluster 4:
['new', 'world', 'battery', 'scientists', 'energy', 'water', 'power',
'researchers', 'cells', 'solar']
```




```
Running Word2Vec-Bag-of-Words with 100 bins

Silhouette Score (bins=100): 0.023186035696326695
Davies-Bouldin Index (bins=100): 4.8610939017468455
Within-Cluster SSE for bins=100: 609.976084777978
Cluster 0:
['new', 'energy', 'world', 'power', 'battery', 'scientists', 'researchers',
'water', 'using', 'cells']
Cluster 1:
['engineers', 'space', 'glass', 'engine', 'mit', 'air', 'treatment', 'vision',
'helps', 'flight']
Cluster 2:
['new', 'world', 'robot', '3d', 'ai', 'scientists', 'researchers', 'bacteria',
'concrete', 'like']
Cluster 3:
['new', 'reactor', 'world', 'robot', 'researchers', 'water', 'scientists',
'breakthrough', 'method', 'control']
Cluster 4:
['new', 'scientists', 'laser', 'battery', 'researchers', 'tech', 'used', 'based',
'cancer', '000']
```




```
Running Word2Vec-Bag-of-Words with 200 bins

Silhouette Score (bins=200): 0.020660138468283074
Davies-Bouldin Index (bins=200): 6.247985976989204
Within-Cluster SSE for bins=200: 661.8948170287945
Cluster 0:
['space', '3d', 'powered', 'flight', 'air', 'engineers', 'glass', 'fuel',
'inspired', 'data']
Cluster 1:
['new', 'researchers', 'energy', 'power', 'scientists', 'world', 'battery',
'cells', 'water', 'ai']
Cluster 2:
['new', 'hydrogen', 'world', 'researchers', 'second', 'scientists', 'tech', 'ai',
'help', 'fuel']
Cluster 3:
['new', 'world', 'ai', 'scientists', '1st', 'create', 'robot', 'million',
'precise', 'color']
Cluster 4:
['new', 'scientists', 'heart', 'researchers', 'robot', 'like', 'world', '99',
'based', 'develop']
```



Among these three configurations, the one with 50 bins has the largest Silhouette Score, smallest DBI and within-cluster SSE. But the clustering boundaries for all these three configurations are not clear, and the high TF-IDF words in each cluster are not semantically close.

Therefore, Doc2Vec is a better method for embedding than Word2Vec in this data set. However, since the data set is relatively small (with only 1,000 posts), the results may be different if the data set is increased.