DSC 478 Final Project
Mike Messina & Omar Khan

# Text Analysis – UFO Intelligence from the CIA

## I. Overview

Since the late 1980s, the CIA had only released about 1,000 pages of classified information regarding unidentified flying objects (UFOs) after a FOIA court case forced the publication of relevant documents. Recently in late December 2020 however due to the Freedom of Information Act, The CIA has declassified upwards of 2,780 documents detailing the agency's findings on unidentified flying objects (UFOs). The contents of these documents range from information regarding UFOs from private citizen sightings that appeared in the media to internal CIA intelligence communications regarding specific case numbers and research into certain phenomena by divisions within the agency. The CIA has claimed that this is their entire collection but with no way to substantiate that claim, it seems appropriate to try and apply machine learning algorithms to parse the declassified pages looking for any patterns of speech that may indicate additional documents still classified or any signs of the agency attempting to hide or muddy the information included in the declassified documents. Our document analysis also has the possibility of uncovering absolutely nothing and could lean towards confirming that the US Government does not have any hard evidence on the existence of UFOs nor their relation to extraterrestrial beings.

## II. Analysis Approach

The documents have already been scanned and converted to searchable PDF files by theblackvault.com who have fought for years with the US government to make classified US agency documents available to the public. Many of the documents are poorly photocopied by hand (most likely on purpose by the CIA) with a number of terms redacted so there is a higher assumed error rate of data collection during the pre-processing stage.

As pure word frequencies are of little importance in this type of analysis, inverse term x document frequency categorization methods will be applied as less used words are assumed to have more importance here than words with the highest frequency in documents talking of specific science and subject matter. As TFxIDF methods can sometimes tune out rarely used words, we'll also be applying other methods to categorizing the contents of the text so as to not lose the importance of rarer words which could hold value here. As we explore the contents of these documents, we will attempt to cluster the terms and documents in an explainable and simple way using KNN clustering method.

On top of finding pure word frequencies across documents, ensuring we have comprehensive and simple class types to categorize the terms in regard to how they are used in the documents will be important. Once we can analyze the numerical qualities of our terms will we have a better idea of how many and what kind of classes we want to classify the terms in to. This is important as we could have misleading results if our classes are not well distinguished.

If more time allows, sentiment analysis and word embedding will be applied to the CIA documents in an attempt to see if we can create our own readable documents based on the characteristics of the CIA documents in an attempt to uncover what may still be kept hidden by

the agency. If we can uncover a common tone and language used by the CIA when producing these records, maybe we can perhaps have a better understanding on what is factual and what is not.

The full dataset used for this analysis is contained in ~350 PDF files which contain 670 pages of previously classified CIA intelligence regarding UFOs. Due to computational constraints, we had to reduce the dataset to this size from originally having 714 PDF files which contained 2,780 documents.

**III. Goals**

Our goals for evaluating this dataset are to attempt to discover document groupings/labels through clustering and to then build a classification model to predict the document groupings we found from clustering. From here, we hope to identify important documents to examine further that may contain important information regarding UFOs as we attempt to efficiently filter through over 2,000 once classified CIA documents that could contain a lot of nothing.

**IV. Analysis**

```
#install module to acquire text from PDF image files
#un comment the below command to install the pdfminer for the first time
#pip install pdfminer.six
from pdfminer.high_level import extract_text
import os
import pandas as pd

n=0
docs = []
for root, dirs, files in os.walk("/Users/mike/DSC478/Project/CIAUFOCD-FULL-CONVERTED"):
    for file in files:
        text = extract_text("/Users/mike/DSC478/Project/CIAUFOCD-FULL-CONVERTED/"+str(file)
        text = text.replace('\n','')
        bad_chars = [';', ':', '!', '*', '\n', '-', '.', '~','/',',']
        for i in bad_chars:
            text = text.replace(i,'')
            docs.append(text)
            n+=1
```

To begin, we had to parse the PDF documents to extract the text into Python. The code block above utilized PDF miner to read through the PDF files in the local directory and extract the readable text from the documents. As the documents are very old and scanned without much care, the text extraction picks up on a lot of unreadable characters and adds its own when it misreads a character. See *exhibit 1* in the Appendix for an example of one of the documents. In order to clean up the text extraction, the line breaks and "bad characters" were replaced with blanks to create a more readable string representation of each document.

After the text from the PDF files has been extracted, we turned to build our document x term matrix using the Count Vectorizer module in sklearn.

| | 00 | 000 | 0000 | 000056961 | 00015 | 0002 | 0005 | 000502337 | 000526282 | 000528415 | ... | ztrs | zunilda | zvezda | zvolen | zvyazd |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 | |
| **1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 | |
| **2** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 | |
| **3** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 | |
| **4** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 | |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | .. |
| **665** | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 | |
| **666** | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 | |
| **667** | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 | |
| **668** | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 | |
| **669** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 | |

670 rows × 18123 columns

As you can see above, we start with 18,123 terms and there are still terms captured that don't mean anything. To further clean and reduce our term features, we'll start by removing terms that begin with zero so that we don't lose any dates but have no need for string representations of integers. In order to further reduce the size of our term feature dimensions, we utilized the Pyenchant python package which includes a spellchecking library using the English dictionary that we can leverage to check for "real english words" in our term feature set. The final “cleaned” doc x term matrix can be seen below. We were able to reduce our term features by about 10,000 and as you can see in the output above, the terms are much more interpretable. The further reduction of our dataset should help combat high assumed high variance in our dataset which should help to prevent some overfitting in our models.

**Doc x Term Matrix**

```
1 dt
```

| | ab | abandon | abandonment | abide | ability | able | abnormal | abnormality | aboard | about | ... | youths | yuan | zero | zigzag | zi |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | ... | 0 | 0 | 0 | 0 | |
| **1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | ... | 0 | 0 | 0 | 0 | |
| **2** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | ... | 0 | 0 | 0 | 0 | |
| **3** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | ... | 0 | 0 | 0 | 0 | |
| **4** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | ... | 0 | 0 | 0 | 0 | |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | |
| **665** | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 2 | 1 | 5 | ... | 0 | 2 | 0 | 0 | |
| **666** | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 2 | 1 | 5 | ... | 0 | 2 | 0 | 0 | |
| **667** | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 2 | 1 | 5 | ... | 0 | 2 | 0 | 0 | |
| **668** | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 2 | 1 | 5 | ... | 0 | 2 | 0 | 0 | |
| **669** | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 2 | 1 | 5 | ... | 0 | 2 | 0 | 0 | |

670 rows × 7681 columns

Next, we turn our attention to normalizing our doc x term matrix into a TFxIDF dataframe. Document frequencies were calculated which allows us to see the 5 most frequent words and the 5 least frequent words below.

| | 0 |
|---|---|
| **for** | 670 |
| **is** | 670 |
| **here** | 670 |
| **of** | 670 |
| **vault** | 670 |
| **...** | ... |
| **vale** | 1 |
| **spreads** | 1 |
| **soc** | 1 |
| **pied** | 1 |
| **wit** | 1 |

7681 rows × 1 columns

The words “for”, “is”, “here”, “of”, “vault” all appear on every document as the theblackvault.com includes a disclaimer on the bottom of each of the documents they obtained from the FBI. If we find our model is sensitive to noise or overfit, we should remove the terms that appear in the disclaimer from our feature space.

The TDxIDF matrix is shown below. In order to cluster on the dataframe, the transpose of the matrix must be taken so that we can have our terms as features and documents as rows.

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | ... | 660 | 661 | 662 | 663 | 664 | 665 | 666 | 667 | 668 | 669 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **ab** | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| **abandon** | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| **abandonment** | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| **abide** | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 5.07 | 5.07 | 5.07 | 5.07 | 5.07 | 5.07 | 5.07 | 5.07 | 5.07 | 5.07 |
| **ability** | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 4.07 | 4.07 | 4.07 | 4.07 | 4.07 | 4.07 | 4.07 | 4.07 | 4.07 | 4.07 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **zine** | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 6.07 | 6.07 | 6.07 | 6.07 | 6.07 | 6.07 | 6.07 | 6.07 | 6.07 | 6.07 |
| **zodiac** | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| **zone** | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 6.13 | 6.13 | 6.13 | 6.13 | 6.13 | 6.13 | 6.13 | 6.13 | 6.13 | 6.13 |
| **zones** | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| **zoomed** | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |

7681 rows × 670 columns

## *Clustering*

We utilized Kmeans module from sklearn to first cluster on our TFxIDF matrix. We initially experimented with 10 clusters but the majority of documents were clustered to one cluster and there was a severe lack of interpretability in some clusters which suggested we were using too many. As such, we went with five clusters as we believe five maximized the explainability of each cluster although five clusters also seemed to cluster the majority (92%) of our documents into one cluster.

```
Size of Cluster 0 =  550
Size of Cluster 1 =  9
Size of Cluster 2 =  5
Size of Cluster 3 =  10
Size of Cluster 4 =  10
Size of Cluster 5 =  10
Size of Cluster 6 =  10
Size of Cluster 7 =  2
Size of Cluster 8 =  4
Size of Cluster 9 =  60
```

```
Size of Cluster 0 =  20
Size of Cluster 1 =  20
Size of Cluster 2 =  10
Size of Cluster 3 =  10
Size of Cluster 4 =  610
```

A quick view of the document clustering is shown to the left. For example, the first 5 documents were clustered into Cluster 4 while the last 5 documents were clustered into Cluster1.

| | Cluster |
|---|---|
| **0** | 4 |
| **1** | 4 |
| **2** | 4 |
| **3** | 4 |
| **4** | 4 |
| **...** | ... |
| **665** | 1 |
| **666** | 1 |
| **667** | 1 |
| **668** | 1 |
| **669** | 1 |

670 rows × 1 columns

From here, we computed the cluster centroids and took the absolute values to try and interpet the terms that are most closely related to each cluster. Cluster 2 is shown below and seems to include documents that reference words that begin with "gu". We can also see that the terms with the farthest distance from this cluster center are all terms with a scientific subject (science, academy, space, mars) perhaps suggesting that a large majority of the CIA documents on UFOs are related to scientific information. You can view the other cluster centroids in *Exhibit 2* in the Appendix.

```
1 centroidAbs.sort_values(by = 2, axis=1, ascending=True)
```

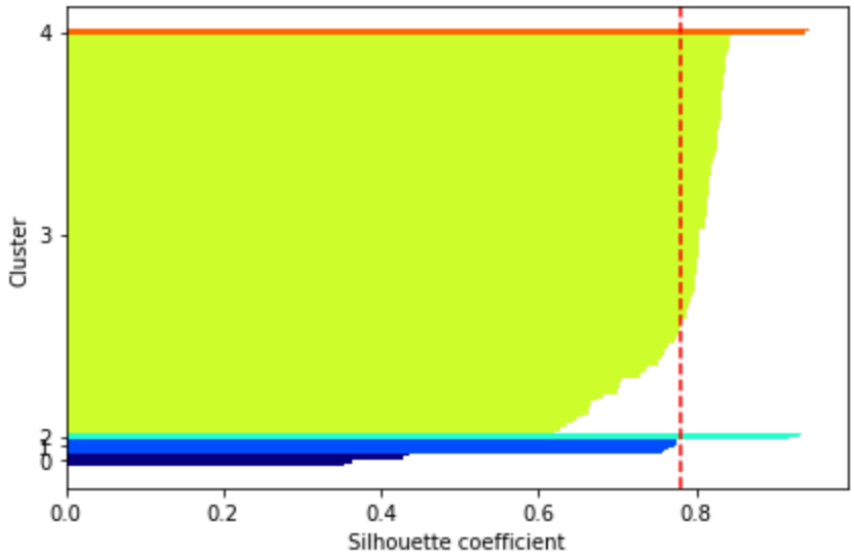
| | zoomed | guerrilla | guerrillas | guest | renounce | guide | guided | renewed | guinea | guise |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 0.00e+00 | 0.00e+00 | 0.00e+00 | 0.00e+00 | 0.00e+00 | 0.00 | 0.00e+00 | 0.00e+00 | 0.00 | 0.00e+00 |
| **1** | 3.03e+00 | 6.07e+00 | 4.16e-17 | 3.03e+00 | 8.33e-17 | 2.24 | 4.16e-17 | 4.16e-17 | 4.48 | 4.16e-17 |
| **2** | 0.00e+00 | 0.00e+00 | 0.00e+00 | 0.00e+00 | 0.00e+00 | 0.00 | 0.00e+00 | 0.00e+00 | 0.00 | 0.00e+00 |

```
1 centroidAbs.sort_values(by = 2, axis=1, ascending=False)
```

| | mars | space | project | science | we | our | scientific | it | academy | are |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 0.00e+00 | 5.43 | 0.00 | 21.72 | 68.14 | 4.36e+01 | 32.76 | 45.09 | 1.02e+01 | 38.06 |
| **1** | 0.00e+00 | 0.91 | 17.53 | 5.79 | 0.73 | 1.78e-15 | 5.96 | 8.37 | 2.22e-16 | 30.14 |
| **2** | 1.55e+02 | 137.60 | 112.99 | 111.50 | 105.72 | 9.77e+01 | 83.40 | 83.18 | 8.13e+01 | 80.71 |

To see how our clustering went, silhouette values from sklearn metrics were run on the TFxIDF matrix and the clusters we found. The average silhouette value from the clustering was noted to be .78. Silhouette value ranges from -1 to 1. An average value of .78 means our documents are actually pretty well matched to their own clusters and separated from other clusters even though the majority of the documents were put into one cluster (cluster 4).

We can view the chart below to see the silhouette values of each cluster. Cluster 0 has the value closest to zero (documents not as well matched to one another) while the other four clusters all have silhouette values around the average of .78 suggesting the documents in these clusters are well matched.




## Classification

We will first build a classifier model using the TFxIDF matrix and compare this to our PCA classifier. The data was split into a 20/80 test/train sample and the neighbors module from

**Measuring Accuracy of Clustering**

```
from sklearn.metrics import classification_report

print(classification_report(target_test, knnpreds_test))
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         3
           1       1.00      1.00      1.00         4
           2       1.00      1.00      1.00         1
           3       1.00      1.00      1.00       123
           4       1.00      1.00      1.00         3

    accuracy                           1.00       134
   macro avg       1.00      1.00      1.00       134
weighted avg       1.00      1.00      1.00       134
```

```
print(knnclf.score(test, target_test))
```

```
1.0
```

```
print(knnclf.score(train, target_train))
```

```
1.0
```

sklearn was utilized to build the classifier with the 5 nearest neighbors. After the model was fit on the train data, we ran predictions on the testing data which consisted of 134 documents.

As shown in the accuracy metrics above, we don't like seeing a perfect accuracy score from our Classification. This is most likely due to our model being highly overfit due to the amount of noise in our dataset from filler and stop words that are affecting the cluster labels. The classifier is being biased by the same things that the clustering was influenced by it seems like. We need to further reduce the term feature space to combat this.

## PCA

As our term feature space is so large and filled with noise, we will attempt to reduce this dimension by computing principal components on the dataset. We'll cluster on this reduced dimensional space and then build a classification model to predict the cluster labels we find. We utilized the sklearn decomposition module to compute the principal components and found four PCs are needed to explain 83% of the variance in our data.

```
1 print(pca.explained_variance_ratio_)
```

```
[0.46 0.23 0.08 0.05 0.04 0.03 0.02 0.01 0.01 0.  ]
```

## Clustering with PCA Data

As we did when we clustered on our TF x IDF dataset, we utilized the Kmeans module in sklearn to perform our clustering. We found that clustering on the PCA matrix yielded the same results where the first five documents were also clustered into the same cluster as well as the last five documents also being clustered into the same cluster.

| | Cluster |
|---|---|
| 0 | 3 |
| 1 | 3 |
| 2 | 3 |
| 3 | 3 |
| 4 | 3 |
| ... | ... |
| 665 | 1 |
| 666 | 1 |
| 667 | 1 |
| 668 | 1 |
| 669 | 1 |

670 rows × 1 columns

We can't infer as much from the cluster centroids on PCA data matrix compared to our TFxIDF matrix as we can't see exact terms that are underlying in each PC. In the dataframe below, we can sort by cluster to see which PCs are related to each cluster.

| | 3 | 8 | 7 | 9 | 5 | 6 | 4 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 78.03 | -4.22 | -4.06 | 1.26e-01 | 7.90 | -4.20 | 169.75 | 287.13 | 292.00 | 28.31 |
| 1 | -0.60 | -0.14 | 0.02 | -1.44e-02 | 1.03 | 8.35 | -0.01 | 1.79 | -104.66 | 994.49 |
| 2 | 6.02 | -2.40 | -2.18 | -2.92e-01 | 0.84 | 2.72 | 4.40 | -342.86 | 835.15 | 146.63 |
| 3 | 2.20 | 0.25 | 0.23 | 1.88e-03 | -0.19 | -0.20 | -1.79 | -8.54 | -24.99 | -36.43 |
| 4 | -294.95 | -4.28 | -3.63 | -4.55e-02 | -6.86 | 1.01 | -234.66 | 285.98 | 314.73 | 30.15 |

Using the "cluster_sizes()" function written below, we can again view the total document assignments to each cluster. We obtained extremely similar results to when we clustered on the TFxIDF matrix where 92% of our documents were clustered into one cluster. Aside from suggesting that there is too much noise in our dataset, we can also determine that these 610 documents could hold the most important information from the other 60 documents and use these clustering assignments as a way to filter out meaningless documents that do not offer much information regarding UFOs.

```
def cluster_sizes(clusters):
    #clusters is an array of cluster labels for each instance in the data

    size = {}
    cluster_labels = np.unique(clusters)
    n_clusters = cluster_labels.shape[0]

    for c in cluster_labels:
        size[c] = len(DTtrans[clusters == c])
    return size
```

```
size = cluster_sizes(clusters)

for c in size.keys():
    print("Size of Cluster", c, "= ", size[c])
```

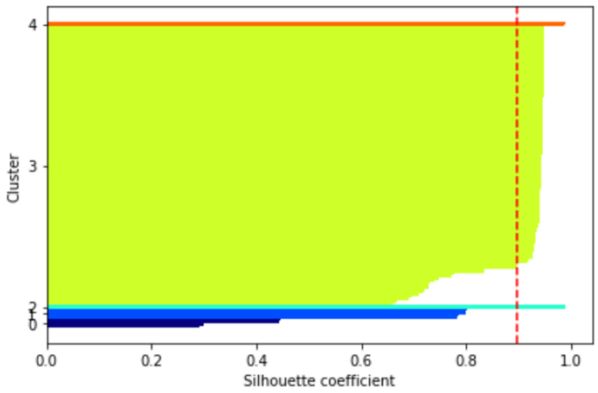
```
Size of Cluster 0 =  20
Size of Cluster 1 =  20
Size of Cluster 2 =  10
Size of Cluster 3 =  610
Size of Cluster 4 =  10
```

To really gauge how our clustering on this PCA dataset went, let's take a look at the average silhouette value to measure the cohesion (intra) and separation (inter) of our clusters. Using the sklearn metrics, we found our average silhouette value to be .896 which is higher than the .78 value we got when we clustered on our TFxIDF values suggesting that our documents are actually better matched to their own clusters and separated from other clusters even though each of the five clusters hold the same amount of documents as our clustering on tf x idf.

```
print(silhouettes.mean())
```

```
0.8967678258847013
```

```
plot_silhouettes(DTtrans, clusters)
```



Classification on PCA Clusters

Again we will use KNN to predict the class labels we found from clustering our PCA data using the neighbors package in sklearn. We'll also again use 5 neighbors so we can compare on a similar scale to our other classification model.

After fitting the model on 80% of our PCA data matrix, we are also getting 100% accuracy in predicting the class labels of each of our documents in the test and train set. While an accuracy measure of 100% is actually not optimal, it's something of value that we can take away that both classification models we have built are overfit due to the size of our term features. We need to further reduce this dimension to also reduce the noise that our models are clustering to.

```
n_neighbors = 5

knnclf = neighbors.KNeighborsClassifier(n_neighbors, weights = 'distance')
knnclf.fit(train, target_train)
```

```
knnpreds_test = knnclf.predict(test)
```

```
print(classification_report(target_test, knnpreds_test))
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         3
           1       1.00      1.00      1.00         4
           2       1.00      1.00      1.00         3
           3       1.00      1.00      1.00         1
           4       1.00      1.00      1.00       123

    accuracy                           1.00       134
   macro avg       1.00      1.00      1.00       134
weighted avg       1.00      1.00      1.00       134
```

```
print(knnclf.score(test, target_test))
```

```
1.0
```

```
print(knnclf.score(train, target_train))
```

```
1.0
```

**V. Conclusion**

We had stated that at the outset our prediction had been that amongst the entirety of all the documents released by the CIA on UFOs only a portion of that documentation would contain any pertinent information. Our goal was to identify those documents specifically and present them as targets for potential future analysis. We conducted Kmeans Clustering and KNN Classification on both TFxIDF and PCA approaches. The end result was inline with our predictions, a large majority of the sampled documents fell into one cluster. Specifically of the

670 documents that were analyzed, a total of 610 fell into one cluster and as such should be deemed relevant for review. A future analysis should be done on the 610 identified documents as the clustering of these documents suggests that they are closely related and could have a higher probability of including important and useful information compared to the other documents not included in the main cluster.

# Appendix

**Exhibit 1**

C00386438

UNCLAS 3S/FAX
*** BEGIN MESSAGE 27 ***
SERIAL=PM1704132591 UDN=X(52912)
CLASS=UNCLAS 3S/FAX
UNCLAS 3S/FAX
SERIAL: PM1704132591
PASS: COPY TO [redacted]
COUNTRY: USSR
SUBJ: DIVERGING OPINIONS ON CAUSE OF SASOVO EXPLOSION
REF: LD1204142191 MOSCOW TASS ENGLISH 121303 -- 'STRANGE ROAR' PRECEDES 'MYSTERIOUS BLAST' IN SASOVO
SOURCE: MOSCOW PRAVDA IN RUSSIAN 15 APR 91 FIRST EDITION P 3
TEXT:
//((CORRESPONDENT N. KIREYEV REPORT: "INCIDENTS: SALTPETER BLOWN UP FROM UFO?"))
((TEXT)) RYAZAN OBLAST -- A HIGHLY POWERFUL EXPLOSION RANG OUT AT MIDNIGHT 12 APRIL NEAR THE SMALL CITY OF SASOVO IN THE RYAZAN AREA. THE SHOCK WAVE ROARED DOWN ITS STREETS, RIPPING OFF ROOFS IN SOME AREAS AND KNOCKING OUT WINDOWS -- EVEN THE FRAMES -- IN A GOOD HALF OF THE APARTMENT BLOCKS, BUILDINGS, AND STRUCTURES. ACCORDING TO EYEWITNESS ACCOUNTS, MULTISTORY BLOCKS ROCKED AS IN AN EARTHQUAKE.
THE ALARMED RESIDENTS COULD NOT RECOVER THEIR COMPOSURE BEFORE MORNING, LOST IN CONJECTURE AS TO WHAT HAD HAPPENED. INCIDENTALLY, NO ONE SUFFERED SERIOUS INJURY ALTHOUGH THERE WERE VISITS TO MEDICAL INSTITUTIONS: SOME PEOPLE WERE IN SHOCK AND SOME HAD BEEN CUT BY BROKEN GLASS...
KINDERGARTENS AND SCHOOLS WERE TEMPORARILY NONOPERATIONAL. DRAFTS BLEW THROUGH BUILDINGS BEFORE THE FRAMES WERE COVERED WITH POLYTHENE FILM.
"THE POPULATION IS NOW LESS WORRIED ABOUT THE RESULTS OF THE INCIDENT," A, ROZHKOV, CHAIRMAN OF SASOVSKIY RAYON SOVIET OF PEOPLE'S DEPUTIES AND PARTY RAYKOM ((RAYON PARTY COMMITTEE)) FIRST SECRETARY, BELIEVES, "THAN BY THE UNCERTAINTY. IT IS STILL UNCLEAR WHAT CAUSED THE EXPLOSION. SOME PEOPLE ARE TALKING ABOUT MUNITIONS LEFT BURIED SINCE THE LAST WAR, WHILE OTHERS CLAIM THAT A POWERFUL AIR BOMB FELL, A THIRD GROUP BLAME IT ON A METEORITE, AND A FOURTH GROUP BLAME UFO'S... THERE ARE PEOPLE WHO SUPPOSEDLY SAW A MOVING FIERY SPHERE."
"MILITARY MEN WHO VISITED THE SCENE OF THE EXPLOSION," ANATOLIY FEDOROVICH WENT ON, "PUT FORWARD THEIR OWN HYPOTHESIS: THEY SAY THAT AMMONIUM NITRATE EXPLODED. WE DO NOT KNOW WHOM AND WHAT TO BELIEVE. ONE THING IS REASSURING: NO VARIATIONS IN THE RADIATION BACKGROUND AND THE COMPOSITION OF THE ATMOSPHERE HAVE BEEN DETECTED. WE ARE WAITING FOR SPECIALISTS FROM THE USSR ACADEMY OF SCIENCES AND FOR THE RESULTS OF A CHEMICAL AND PHYSICAL ANALYSIS."
CERTAINLY, THE CAUSE OF THE EXPLOSION IS STILL A MYSTERY ALTHOUGH THE CRATER 1 KM FROM THE CITY, 28 METERS IN DIAMETER, AND ROUGHLY 4 METERS DEEP IS, REGRETTABLY, A REALITY. ALONGSIDE THERE WERE 30 TONNES OF FRESH AMMONIUM NITRATE WHICH WAS BROUGHT TO THE FIELD 5 APRIL TO FERTILIZE THE PLANTS. BUT CROP FARMERS HAVE, AFTER ALL, SUCCESSFULLY USED THIS FERTILIZER FOR SEVERAL YEARS NOW THROUGHOUT THE COUNTRY AND ABROAD. LOCAL AGRICULTURAL CHEMISTRY WAREHOUSES CONTAIN 10,000 TONNES OF SALTPETER FROM THE CONSIGNMENT
UNCLAS 3S/FAX

Approved for Release
Date MAY 2000
(b)(3) 54

**Exhibit 2**

*Cluster 0*

| | zoomed | pub | pt | fanaticism | fantasia | psychiatric | psis | farce | farm | farmers |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 0.00e+00 | 0.00e+00 | 0.00 | 0.00e+00 | 0.00e+00 | 0.00e+00 | 0.00e+00 | 0.00e+00 | 0.00e+00 | 0.00 |

*Cluster 1*

| | online | town | towards | totally | inspection | tor | centrist | prove | certainly | cg |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 0.0 | 2.24 | 0.00 | 3.74 | 0.00 | 1.52 | 0.00 | 0.00 | 0.00 | 0.0 |
| **1** | 0.0 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.0 |

*Cluster 2*

| | zoomed | guerrilla | guerrillas | guest | renounce | guide | guided | renewed | guinea | guise |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 0.00e+00 | 0.00e+00 | 0.00e+00 | 0.00e+00 | 0.00e+00 | 0.00 | 0.00e+00 | 0.00e+00 | 0.00 | 0.00e+00 |
| **1** | 3.03e+00 | 6.07e+00 | 4.16e-17 | 3.03e+00 | 8.33e-17 | 2.24 | 4.16e-17 | 4.16e-17 | 4.48 | 4.16e-17 |
| **2** | 0.00e+00 | 0.00e+00 | 0.00e+00 | 0.00e+00 | 0.00e+00 | 0.00 | 0.00e+00 | 0.00e+00 | 0.00 | 0.00e+00 |

*Cluster 3*

| | zoomed | grams | grand | grandfather | grandiose | grandson | granting | grass | resist | gravest |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 0.00e+00 | 0.00e+00 | 0.00 | 3.03e+00 | 0.00 | 3.03e+00 | 0.00e+00 | 3.03e+00 | 0.00e+00 | 0.00e+00 |
| **1** | 3.03e+00 | 4.16e-17 | 2.24 | 4.16e-17 | 0.00 | 4.16e-17 | 4.16e-17 | 4.16e-17 | 3.03e+00 | 4.16e-17 |
| **2** | 0.00e+00 | 0.00e+00 | 0.00 | 0.00e+00 | 4.48 | 0.00e+00 | 6.07e+00 | 0.00e+00 | 0.00e+00 | 0.00e+00 |
| **3** | 0.00e+00 | 0.00e+00 | 0.00 | 0.00e+00 | 0.00 | 0.00e+00 | 0.00e+00 | 0.00e+00 | 0.00e+00 | 0.00e+00 |

*Cluster 4*

| | at | healthy | preparation | biggest | black | clearinghouse | guiding | somewhere | principles | wind |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 0.0 | 2.24 | 0.00 | 0.00 | 0.0 | 0.0 | 4.48 | 4.48 | 1.42e+01 | 2.24 |
| **1** | 0.0 | 4.48 | 2.24 | 6.10 | 0.0 | 0.0 | 0.00 | 0.00 | 1.11e-16 | 0.00 |
| **2** | 0.0 | 0.00 | 8.96 | 4.07 | 0.0 | 0.0 | 4.48 | 4.48 | 4.07e+00 | 8.96 |
| **3** | 0.0 | 4.48 | 4.48 | 4.07 | 0.0 | 0.0 | 4.48 | 4.48 | 8.13e+00 | 4.48 |
| **4** | 0.0 | 0.00 | 0.00 | 0.00 | 0.0 | 0.0 | 0.00 | 0.00 | 0.00e+00 | 0.00 |