Dimension Reduction

CS 780/880 Natural Language Processing Lecture 6

Samuel Carton, University of New Hampshire

Last lecture

Key idea: Clustering text

Concepts

- Unsupervised learning
- Clustering
- K-means clustering
- Clustering metrics
  - Extrinsic
    - Mutual information
  - Intrinsic
    - Silhouette score
- Representing induvial clusters

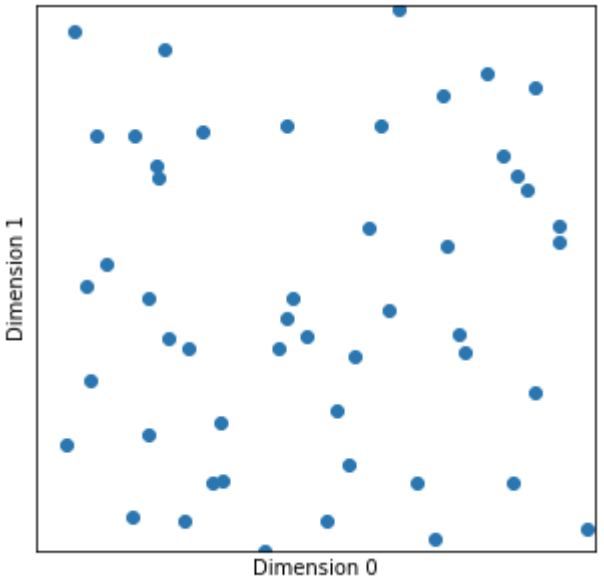
Toolkits

- Matplotlib for data visualization
- Scikit-Learn for model building and evaluation

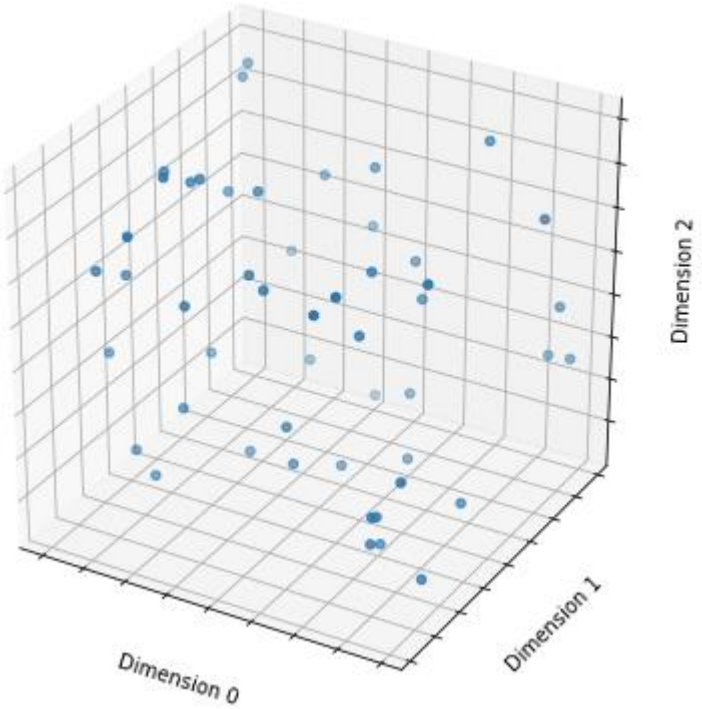
Dimensionality of data

Basic idea: when every item in the dataset consists of N elements, we can think of it as N-dimensional and as therefore points in N-dimensional space

2-dimensional data



3-dimensional data

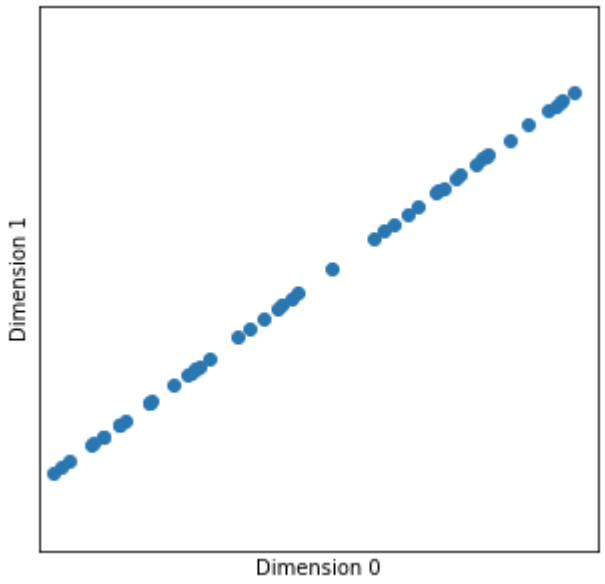


What is the dimensionality of our vectorized text data?

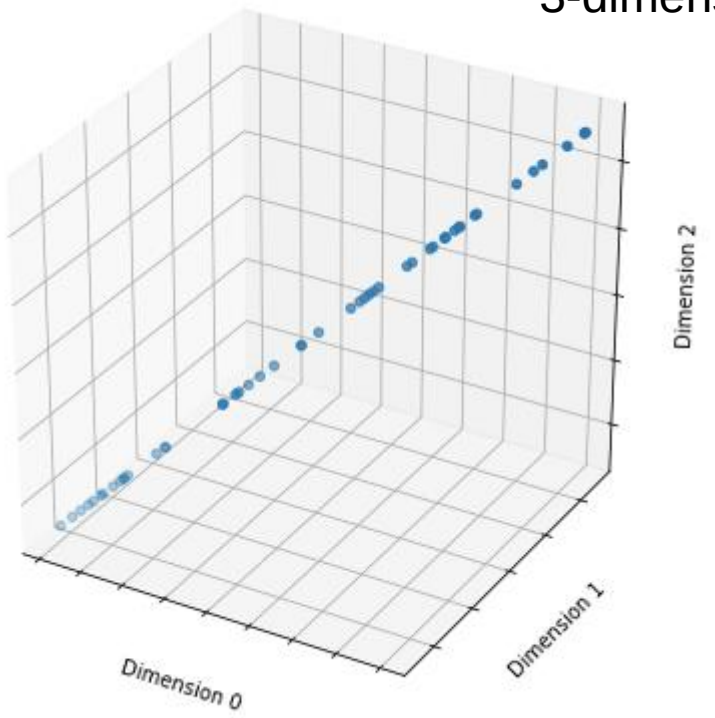
Dimension reduction

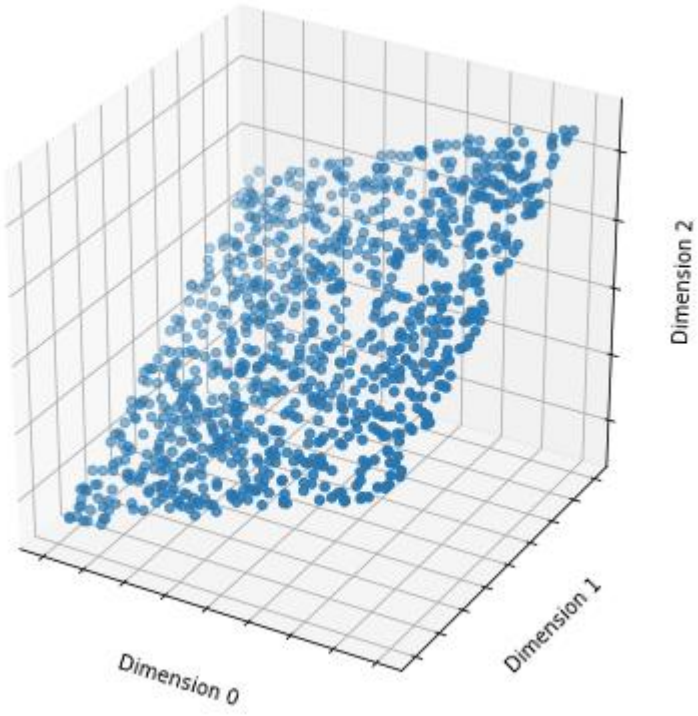
Question: Our text is N-dimensional… but do actually need N dimensions to represent it?

2-dimensional data



3-dimensional data

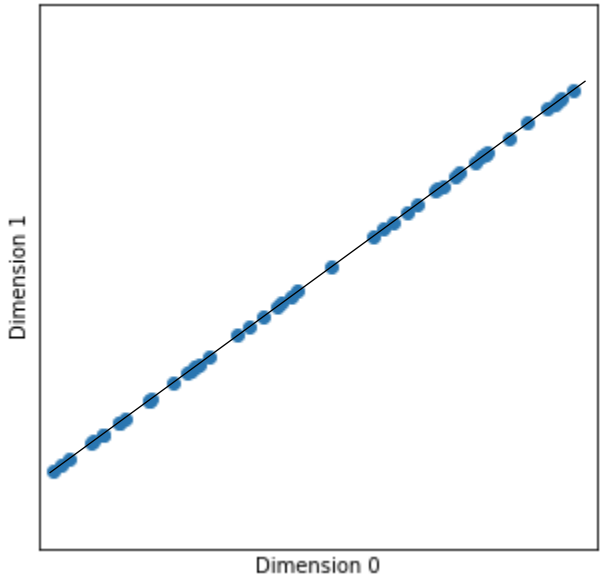


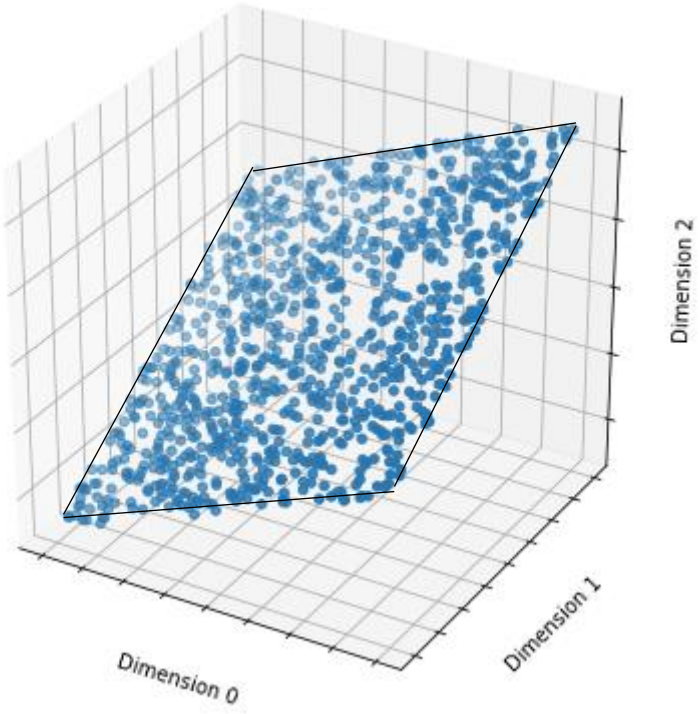


Dimension reduction

Answer: Nope. If we could figure out the line (or plane) along which the data is laid out, we would only need 1 (or 2) values to describe every data point—how far it is along that direction(s)

- I.e. direction of maximum **variance**

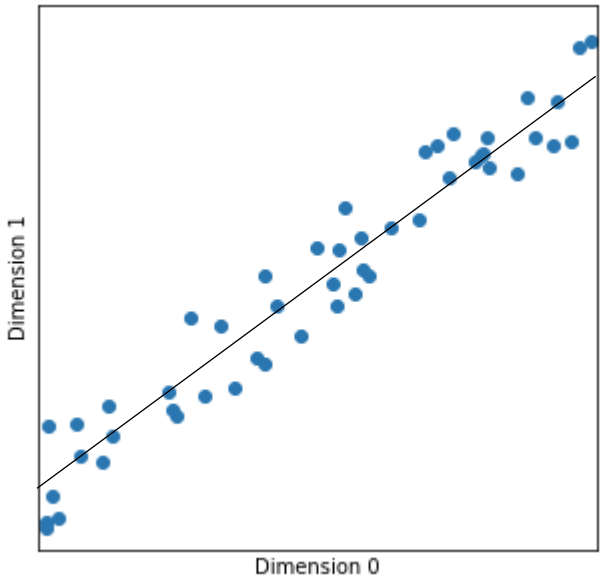


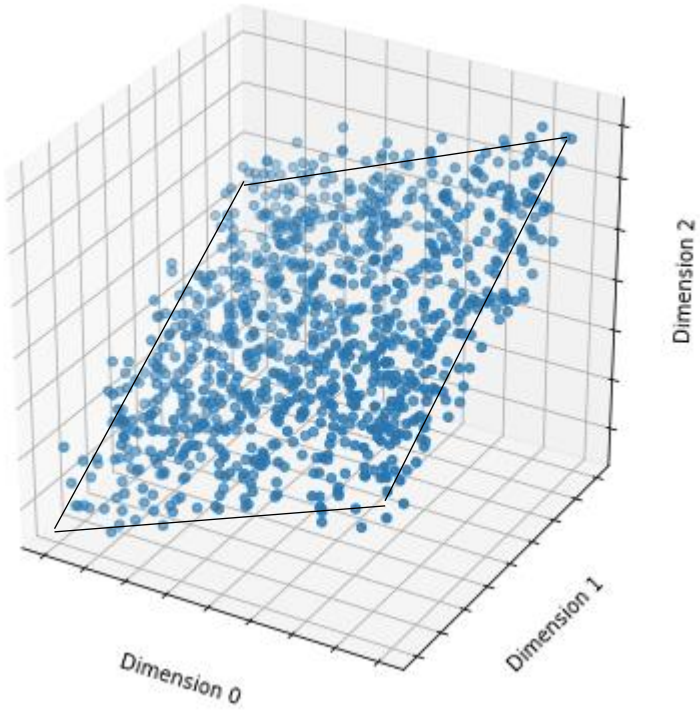


Dimension reduction

But what if the data isn't perfectly laid out across a lower-dimension set of axes?

- Then maybe we can still capture **most** of the variance in the data, such that what is left isn't really that important





Dimension reduction

Basic idea: take a data matrix of size M×N and compress it to M×D, where D << M, while still retaining most of the useful information in the matrix

- For text, go from M *sparse* vectors of dimensionality V=size of the vocabulary, to M *dense* vectors of size D, where D is significantly smaller (100, 200, etc.)
- While still being useful for classification, clustering, etc.

Two most popular approaches:

- Principal component analysis (PCA)
- Singular value decomposition (SVD)

Matrix multiplication

When you multiply two matrixes $A^{m \times n} * B^{n \times p}$, you get $C^{m \times p}$ by calculating the dot product of every row of A with every column of B

Only matrixes with matching inner dimensions (e.g. m×n versus n×p) can be multiplied

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix} \qquad \mathbf{B} = \begin{pmatrix} b_{11} & b_{12} & \cdots & b_{1p} \\ b_{21} & b_{22} & \cdots & b_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ b_{n1} & b_{n2} & \cdots & b_{np} \end{pmatrix}$$

$$\mathbf{AB} = \mathbf{C} = \begin{pmatrix} a_{11}b_{11} + \cdots + a_{1n}b_{n1} & a_{11}b_{12} + \cdots + a_{1n}b_{n2} & \cdots & a_{11}b_{1p} + \cdots + a_{1n}b_{np} \\ a_{21}b_{11} + \cdots + a_{2n}b_{n1} & a_{21}b_{12} + \cdots + a_{2n}b_{n2} & \cdots & a_{21}b_{1p} + \cdots + a_{2n}b_{np} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1}b_{11} + \cdots + a_{mn}b_{n1} & a_{m1}b_{12} + \cdots + a_{mn}b_{n2} & \cdots & a_{m1}b_{1p} + \cdots + a_{mn}b_{np} \end{pmatrix}$$

Matrix transpose

Transposing a matrix, denoted by M^T, switches its dimensions.

Very common operation in linear algebra

Also the only way you can multiply a matrix by itself unless it is square

Examples:

$$\begin{bmatrix} 1 & 2 \end{bmatrix}^{\mathrm{T}} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}^{\mathrm{T}} = \begin{bmatrix} 1 & 3 \\ 2 & 4 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix}^{\mathrm{T}} = \begin{bmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{bmatrix}$$

Covariance matrix

The **covariance matrix** quantifies the joint variability between two random variables *X* and *Y*

$$cov(X, Y) = \mathbb{E}[(X - \mathbb{E}[X])(Y - \mathbb{E}[Y])]$$

When X is a data matrix of n samples × m features, centered on 0, then the covariance matrix can be calculated as:

$$\mathbf{C} = \frac{\mathbf{X}^\top \mathbf{X}}{n - 1}$$

PCA and SVD

Principle component analysis (PCA) and singular value decomposition (SVD) are both **matrix factorization** techniques, which calculate how to express the covariance matrix as a product of lower-dimensionality matrices.

PCA performs an **eigendecomposition** of the covariance matrix C, which learns to represent it as $C = W\Lambda W^{-1}$, where W is a m×m matrix of **eigenvectors**, and Λ a diagonal m×m matrix of **eigenvalues**

SVD performs a decomposition of C into a product of two unitary matrices (U, V*) and a rectangular diagonal matrix of singular values (Σ): $C = U\Sigma V^*$

In both cases, we can get back a lower-dimension approximation of our original **X** by performing the product with subsets of the factor matrices:

Example for PCA: $\mathbf{X}_k = \mathbf{X}\mathbf{W}_k$

PCA vs. SVD in practice

In practice, PCA is calculated “under the hood” using SVD (truncated SVD for text)

- SVD singular values are easily convertible to PCA eigenvalues

So mostly you will be using SVD for dimension reduction in practice

And you can look at the singular values to get a sense of how much of the variance of the original data are explained in the D dimensions you’ve chosen to retain.

Conventional clustering

```
our_categories= [
    "alt.atheism",
    "talk.religion.misc",
    "comp.graphics",
    "sci.space",

]

ng_dataset = fetch_20newsgroups(remove=('headers', 'footers', 'quotes'),
                                categories=our_categories,
                                subset="all",
                                shuffle=True,
                                random_state=42,
                                )
```

```
ng_df = pd.DataFrame({'text':ng_dataset.data, 'group':ng_dataset.target})
ng_df
```

	text	group
0	My point is that you set up your views as the ...	0
1	\nBy '8 grey level images' you mean 8 items of...	1
2	FIRST ANNUAL PHIGS USER GROUP CONFERENCE\n\n ...	1
3	I responded to Jim's other articles today, but...	3
4	\nWell, I am placing a file at my ftp today th...	1
...	...	...
3382	I am working on a program to display 3d wirefr...	1
3383	\n Did the Russian spacecraft(s) on the ill-f...	2
3384	\n\nOh gee, a billion dollars! That'd be just...	2
3385	I am looking for software to run on my brand n...	1
3386	Within the next several months I'll be looking...	1

3387 rows × 2 columns

Conventional clustering

```
model = KMeans(n_clusters=4,
               max_iter=100,
               n_init=5).fit(ng_X)

ng_df['cluster'] = model.predict(ng_X)
ng_df
```

	text	group	preprocessed	cluster
0	My point is that you set up your views as the ...	0	my point is that you set up your view as the o...	1
1	\nBy '8 grey level images' you mean 8 items of...	1	by ' 8 grey level imag ' you mean 8 item of 1b...	2
2	FIRST ANNUAL PHIGS USER GROUP CONFERENCE\n\n ...	1	first annual phig user group confer the first ...	0
3	I responded to Jim's other articles today, but...	3	i respond to jim 's other articl today , but i...	0
4	\nWell, I am placing a file at my ftp today th...	1	well , i am place a file at my ftp today that ...	2
...	...	...	...	...
3382	I am working on a program to display 3d wirefr...	1	i am work on a program to display 3d wirefram ...	2
3383	\n Did the Russian spacecraft(s) on the ill-f...	2	did the russian spacecraft (s) on the ill-fa...	2
3384	\n\nOh gee, a billion dollars! That'd be just...	2	oh gee , a billion dollar ! that 'd be just ab...	0
3385	I am looking for software to run on my brand n...	1	i am look for softwar to run on my brand new t...	2
3386	Within the next several months I'll be looking...	1	within the next sever month i 'll be look for ...	2

3387 rows × 4 columns

Conventional clustering

```python
from sklearn.metrics import normalized_mutual_info_score, adjusted_rand_score, silhouette_score
```

```python
def evaluate_clustering(X, true_labels, cluster_assignments):
  print(f"Normalized mutual information between true labels and cluster assignments: \
  {normalized_mutual_info_score(true_labels, cluster_assignments):.3f}")
  print(f"Adjusted Rand score between true labels and cluster assignments: \
  {adjusted_rand_score(true_labels, cluster_assignments):.3f}")
  print(f"Average silhouette score of cluster assignments: \
  {silhouette_score(X, cluster_assignments, sample_size=2000):.3f}")

```

```python
evaluate_clustering(ng_X, ng_df['group'], ng_df['cluster'])
```

```
Normalized mutual information between true labels and cluster assignments: 0.347
Adjusted Rand score between true labels and cluster assignments: 0.191
Average silhouette score of cluster assignments: 0.009
```

Average silhouette score of cluster assignments: 0.009

SVD for clustering

```
1 from sklearn.decomposition import TruncatedSVD
2 from sklearn.pipeline import make_pipeline
3 from sklearn.preprocessing import Normalizer
```

```
1 pipeline = make_pipeline(TruncatedSVD(n_components=100), Normalizer(copy=False))
2
3 pipeline.fit(ng_X)
```

```
Pipeline(steps=[('truncatedsvd', TruncatedSVD(n_components=100)),
                ('normalizer', Normalizer(copy=False))])
```

SVD for clustering

```
1 print(pipeline[0].explained_variance_ratio_)
```

```
[0.00584104 0.0095496  0.00653745 0.00486296 0.00444869 0.00418118
 0.00406312 0.00366628 0.00364483 0.00347826 0.0031842  0.00308523
 0.00297885 0.00293872 0.00281783 0.00278395 0.00266414 0.00263875
 0.00255509 0.00251268 0.00250039 0.00247195 0.00244391 0.00239149
 0.00237247 0.00234858 0.0023194  0.00229551 0.00225731 0.0022259
 0.0022006  0.00215427 0.0021362  0.00212559 0.00208279 0.00206926
 0.00206116 0.00205015 0.00202851 0.00200864 0.00197045 0.00197029
 0.00196684 0.00192687 0.00191495 0.00189608 0.0018912  0.0018752
 0.00184478 0.00183305 0.0018195  0.00181102 0.00178057 0.00177471
 0.00176546 0.00175936 0.00174562 0.00173275 0.00170926 0.00170442
 0.00169418 0.00167836 0.0016533  0.00164725 0.00164349 0.00163075
 0.00161503 0.00159829 0.00158777 0.00157845 0.00157376 0.00155967
 0.00155458 0.00154128 0.00153191 0.00152127 0.00152078 0.00151378
 0.00149234 0.0014815  0.00147753 0.0014712  0.00145651 0.00144281
 0.00143771 0.00143188 0.00142375 0.00141679 0.00140436 0.00139205
 0.00138605 0.00136832 0.00136518 0.00135738 0.00134907 0.00133329
 0.00132955 0.00130884 0.00130752 0.00129157]
```

```
1 print(pipeline[0].explained_variance_ratio_.sum())
```

```
0.219038407432514
```

SVD for clustering

```
1 r_ng_X = pipeline.transform(ng_X)
2 r_model = KMeans(n_clusters=5, random_state=0).fit(r_ng_X)
3 ng_df['r_cluster'] = r_model.predict(r_ng_X)
4 ng_df
```

	text	group	preprocessed	cluster	r_cluster
0	My point is that you set up your views as the ...	0	my point is that you set up your view as the o...	1	2
1	\nBy '8 grey level images' you mean 8 items of...	1	by ' 8 grey level imag ' you mean 8 item of 1b...	2	0
2	FIRST ANNUAL PHIGS USER GROUP CONFERENCE\n\n ...	1	first annual phig user group confer the first ...	0	0
3	I responded to Jim's other articles today, but...	3	i respond to jim 's other articl today , but i...	0	2
4	\nWell, I am placing a file at my ftp today th...	1	well , i am place a file at my ftp today that ...	2	0
...	...	...	...	...	...
3382	I am working on a program to display 3d wirefr...	1	i am work on a program to display 3d wirefram ...	2	0
3383	\n Did the Russian spacecraft(s) on the ill-f...	2	did the russian spacecraft (s) on the ill-fa...	2	0
3384	\n\nOh gee, a billion dollars! That'd be just...	2	oh gee , a billion dollar ! that 'd be just ab...	0	3
3385	I am looking for software to run on my brand n...	1	i am look for softwar to run on my brand new t...	2	0
3386	Within the next several months I'll be looking...	1	within the next sever month i 'll be look for ...	2	0

3387 rows × 5 columns

SVD for clustering

```
1 print('Evaluation of dimension-reduced clustering:')
2 evaluate_clustering(r_ng_X, ng_df['group'], ng_df['r_cluster'])
3
4 print('\nEvaluation of original clustering:')
5 evaluate_clustering(ng_X, ng_df['group'], ng_df['cluster'])
```

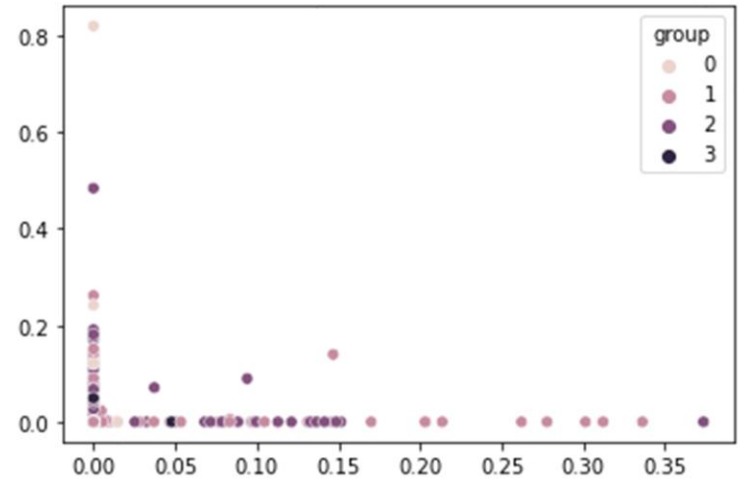
```
Evaluation of dimension-reduced clustering:
Normalized mutual information between true labels and cluster assignments: 0.363
Adjusted Rand score between true labels and cluster assignments: 0.304
Average silhouette score of cluster assignments: 0.030

Evaluation of original clustering:
Normalized mutual information between true labels and cluster assignments: 0.347
Adjusted Rand score between true labels and cluster assignments: 0.191
Average silhouette score of cluster assignments: 0.008
```

SVD for visualizing clusters

```
# Each dimension in the original term-document matrix refers to the presence
# or absence of one word from the vocabulary
# So trying to plot the data in terms of any two of these dimensions is not very
# helpful
dense_ng_X = np.array(ng_X.todense())
sns.scatterplot(x=dense_ng_X[:,0], y=dense_ng_X[:,1], hue=ng_df['group'])
```

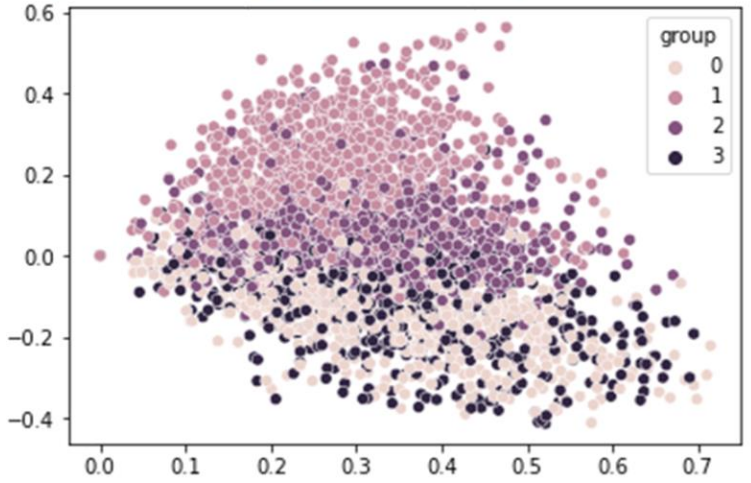
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1cf8d2a580>
```



SVD for visualizing clusters

```
1 # If we instead plot the data in terms of the first two principle
2 # components, we start seeing some real structure
3
4 sns.scatterplot(x=r_ng_X[:,0], y=r_ng_X[:,1], hue=ng_df['group'])
```

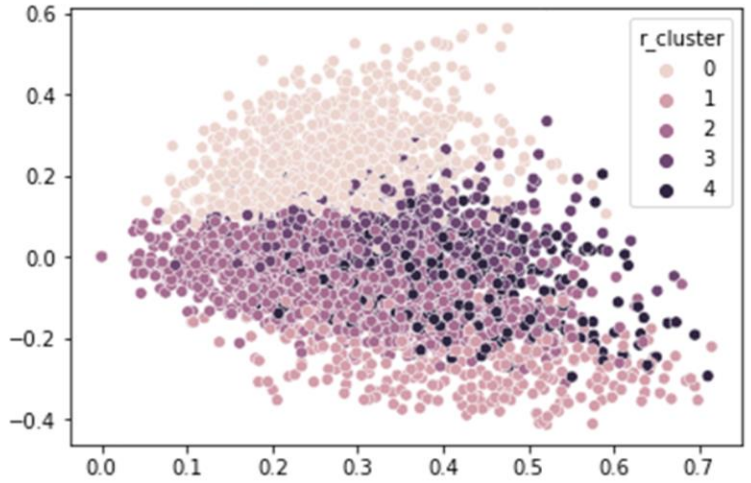
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1cfb2b9280>
```



SVD for visualizing clusters

```
1 # And that structure gets captured by running K-means clustering on that
2 # dimension-reduced data
3
4 sns.scatterplot(x=r_ng_X[:,0], y=r_ng_X[:,1], hue=ng_df['r_cluster'])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1cf8c10ca0>
```

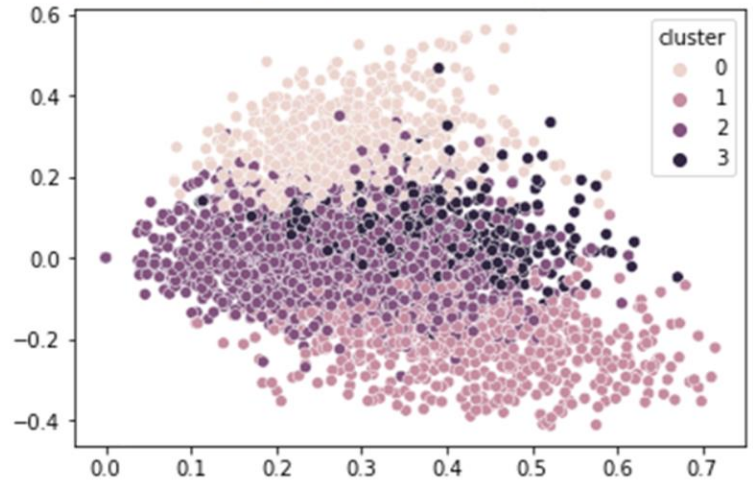


SVD for visualizing clusters

```
# And even doing clustering on the original sparse matrics captured much
# of that same structure, it's just that it was hard to visualize

sns.scatterplot(x=r_ng_X[:,0], y=r_ng_X[:,1], hue=ng_df['cluster'])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1cf8be39a0>
```



Concluding thoughts

In NLP, learning **dense representations** of text is absolutely critical.

You can only get so far with sparse bag-of-words or TF-IDF representations.

Deep learning, aka **representation learning**

- Learns “targeted” dense representations optimized for specific tasks

Dimension reduction: “general-purpose” representations optimized for mathematical properties

- SVD on term-document matrix is called **Latent Semantic Analysis** (1988)

Still useful for prediction, clustering, visualization, etc.

Much of this lecture borrowed from: https://towardsdatascience.com/pca-and-svd-explained-with-numpy-5d13b0d2a4d8