Social Topic Distributions

Hakan Akyürek

Mürüvvet Hasanbaşoğlu

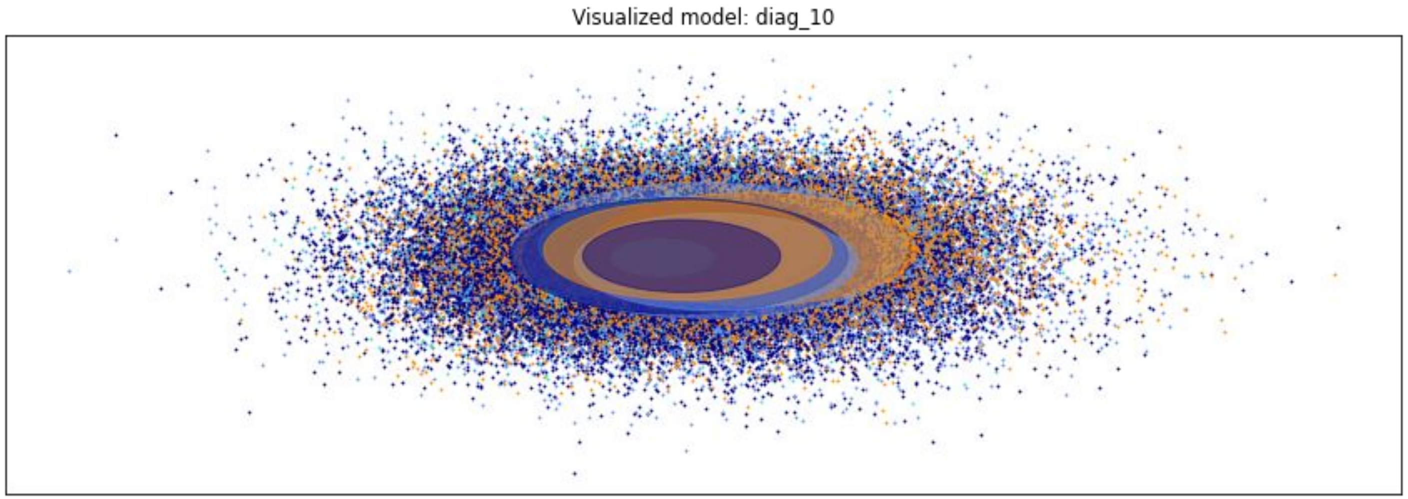
June 2, 2020

Overview

1. **Cluster labeling**
2. **Some more GMM training**
3. **Jensen Shannon Distance**
4. **Next Tasks -** for the next 2 weeks

1. Cluster Labeling

- Gaussians in the model overlap... a lot.
- Some words are close to huge number of the gaussian centroids.
- Empty strings exist in the dataset.
    - It’s embedding is really close to zero point.
- No meaningful labeling!



- <empty-string>, trumpster, illuminati
- <empty-string>, illuminati, malta
- <empty-string>, trumpster, blasting

1. Cluster Labeling

Cluster labeling done in two steps:

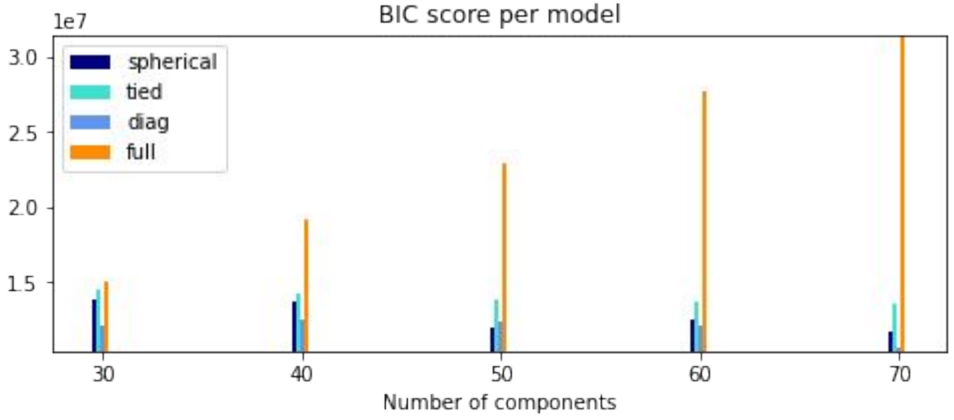
1. Predict a label for each of the words in the vocabulary.
2. For each gaussian check the closest words that were labeled that gaussian's centroid.
   a. Do not check all data points in the space for each gaussian.

Some examples of labeled clusters:

- multicolored, peacock, adorn, jewels
- dimwits, simpletons, interlopers
- racoon, hedgehogs, squirrel
- haha, lol, lolol, anyhoo, ahaha

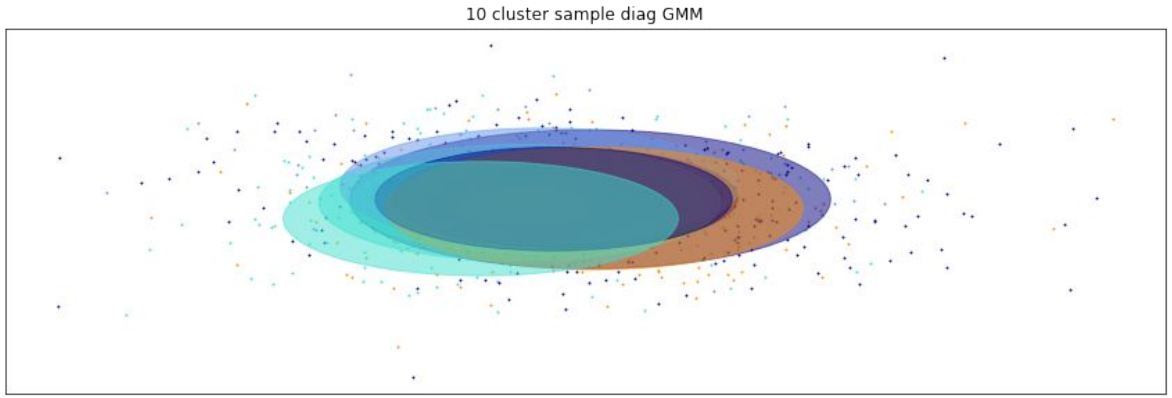
2. Some more GMM training

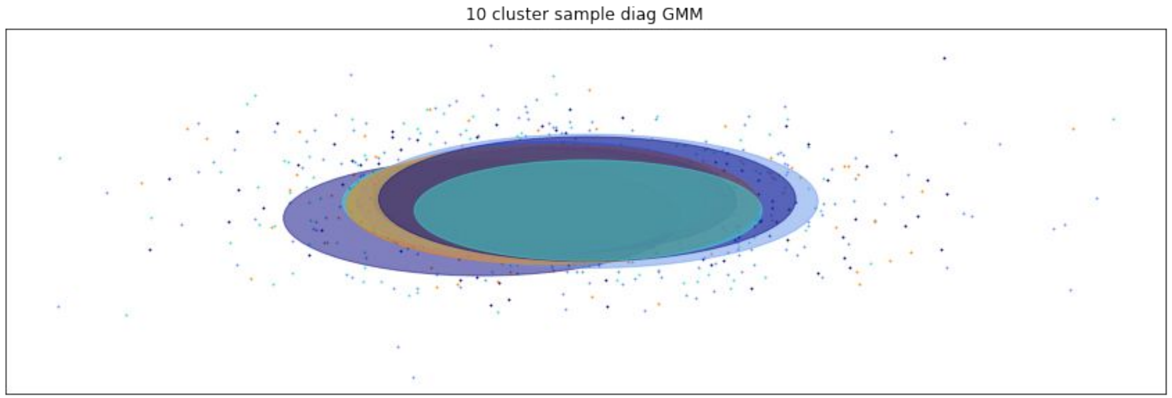
- In last sprint:
    - covariance types = ‘spherical’, ‘full’, ‘diag’, ‘tied’
    - n_components range: 30, 40, … , 70



❖ Selected **‘diag’** covariance type but what is the right n_component actually?

❖ Same model fit with the same data but produced different distributions:



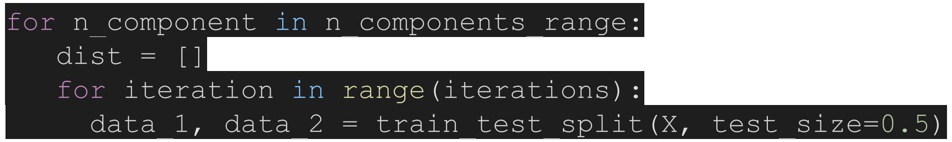


➔ More automated method for finding the **right n_components**!

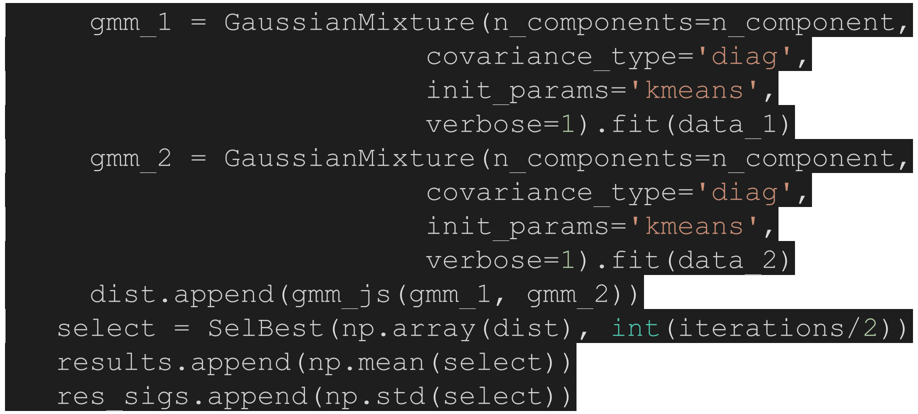
2. Some more GMM training

- n_components range: 2, 4, 6, …, 18, 20, 30, …, 70
- Fit models 10 times with same configuration per component
- Select the best 5 out of 10

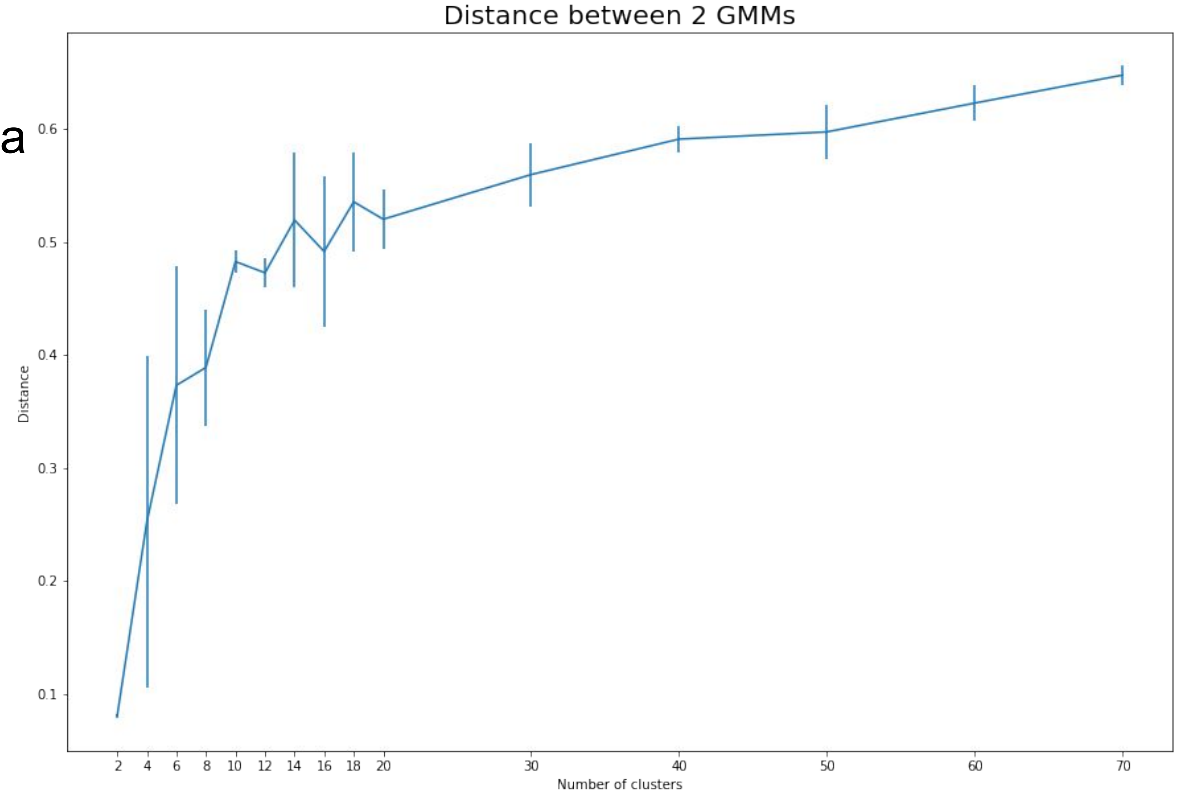
➔ GMMs distance check per cluster with random data



```
for n_component in n_components_range:
   dist = []
   for iteration in range(iterations):
     data_1, data_2 = train_test_split(X, test_size=0.5)
```



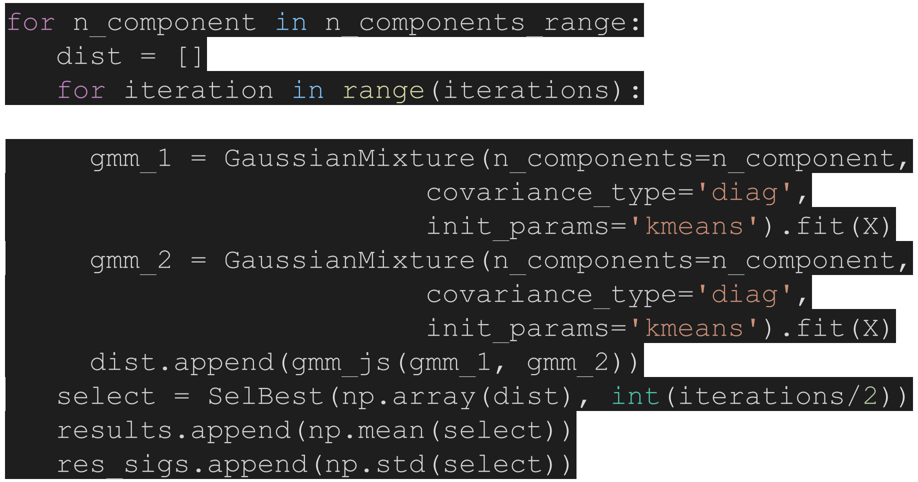
```
     gmm_1 = GaussianMixture(n_components=n_component,
                         covariance_type='diag',
                         init_params='kmeans',
                         verbose=1).fit(data_1)
     gmm_2 = GaussianMixture(n_components=n_component,
                         covariance_type='diag',
                         init_params='kmeans',
                         verbose=1).fit(data_2)
     dist.append(gmm_js(gmm_1, gmm_2))
   select = SelBest(np.array(dist), int(iterations/2))
   results.append(np.mean(select))
   res_sigs.append(np.std(select))
```



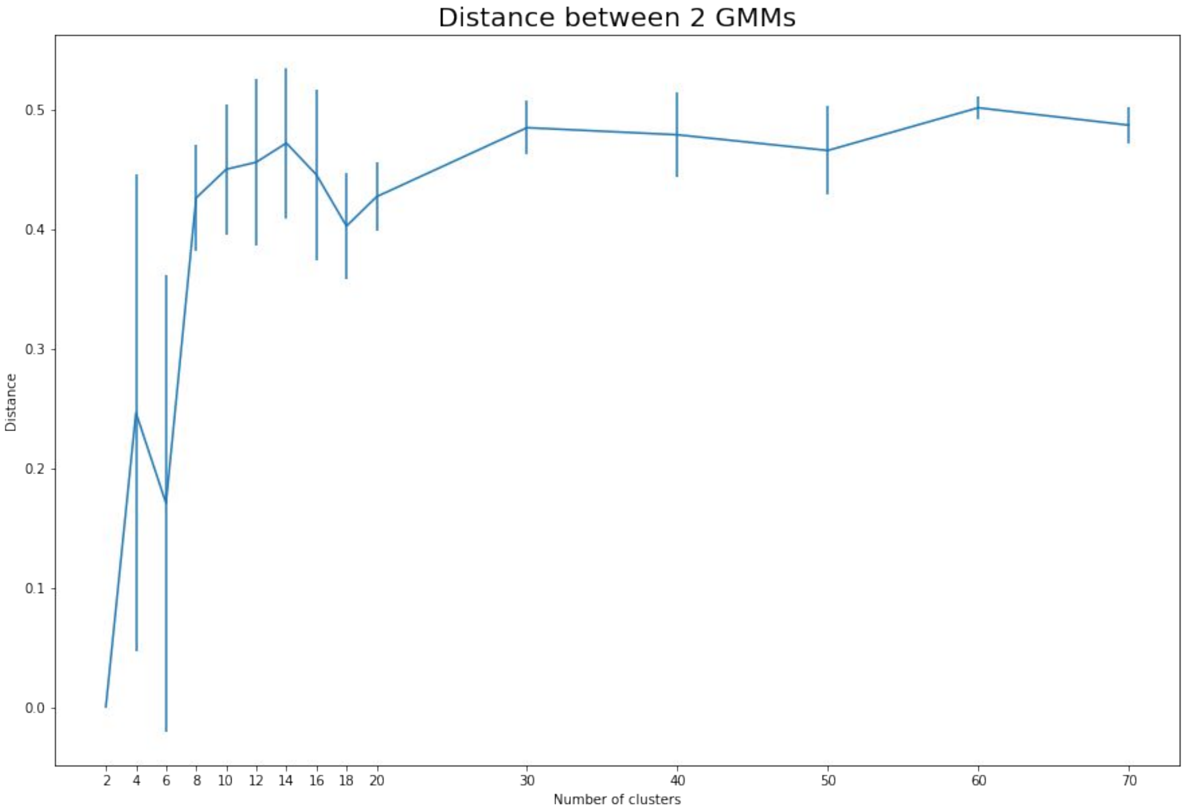
2. Some more GMM training

- n_components range: 2, 4, 6, …, 18, 20, 30, …, 70
- Fit models 10 times with same configuration per component
- Select the best 5 out of 10

➔ GMMs distance check per cluster with same data



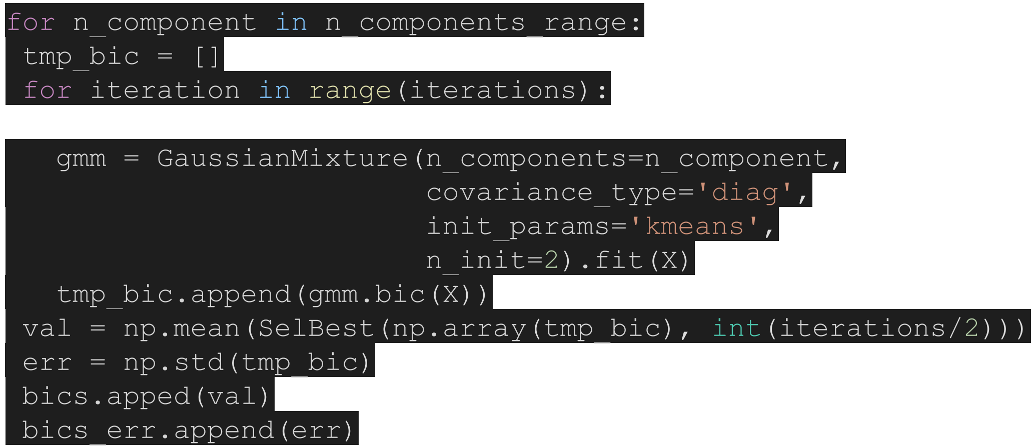
```
for n_component in n_components_range:
    dist = []
    for iteration in range(iterations):

        gmm_1 = GaussianMixture(n_components=n_component,
                                covariance_type='diag',
                                init_params='kmeans').fit(X)
        gmm_2 = GaussianMixture(n_components=n_component,
                                covariance_type='diag',
                                init_params='kmeans').fit(X)
        dist.append(gmm_js(gmm_1, gmm_2))
    select = SelBest(np.array(dist), int(iterations/2))
    results.append(np.mean(select))
    res_sigs.append(np.std(select))
```



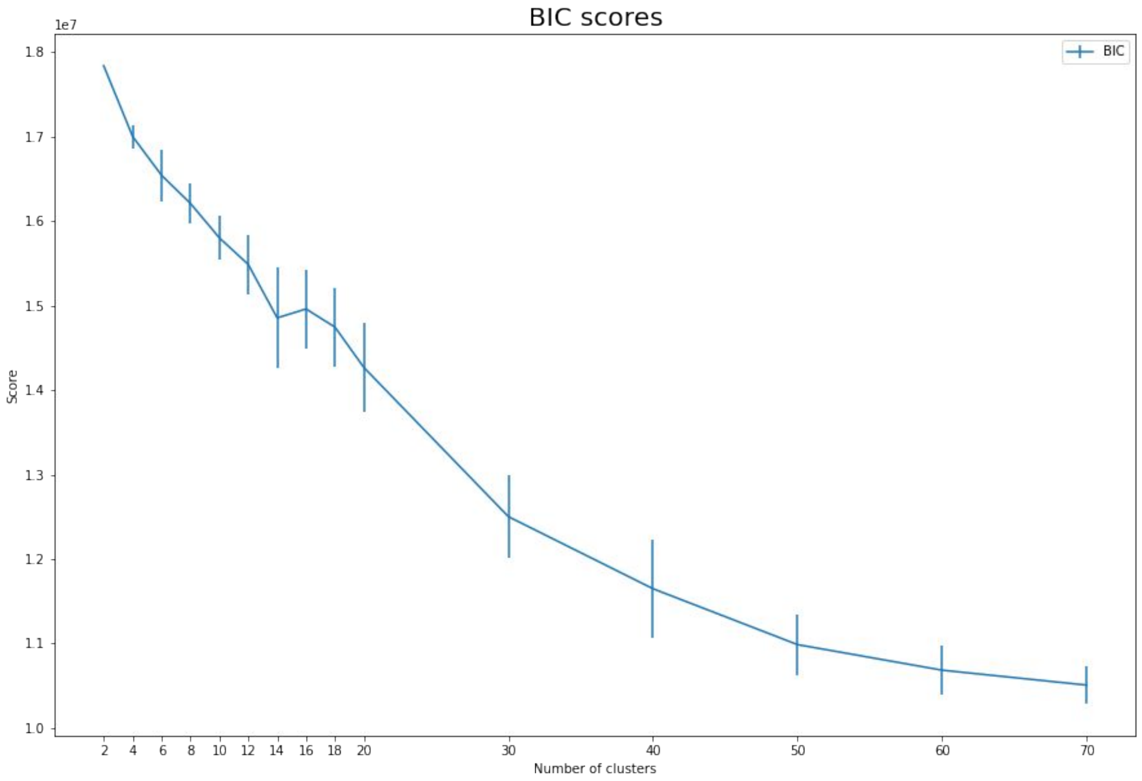
2. Some more GMM training

- n_components range: 2, 4, 6, …, 18, 20, 30, …, 70
- Fit models 10 times with same configuration per component
- Select the best 5 out of 10

➔ BIC score



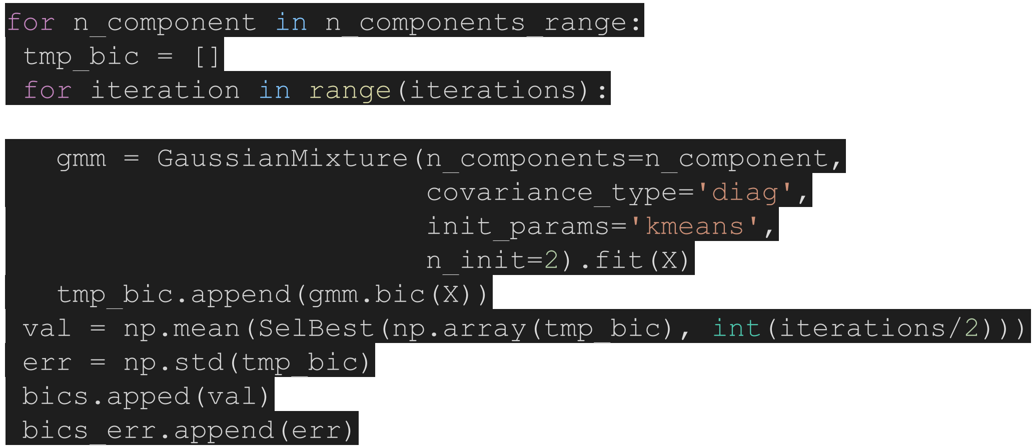
```
for n_component in n_components_range:
 tmp_bic = []
 for iteration in range(iterations):

   gmm = GaussianMixture(n_components=n_component,
                         covariance_type='diag',
                         init_params='kmeans',
                         n_init=2).fit(X)
   tmp_bic.append(gmm.bic(X))
 val = np.mean(SelBest(np.array(tmp_bic), int(iterations/2)))
 err = np.std(tmp_bic)
 bics.apped(val)
 bics_err.append(err)
```



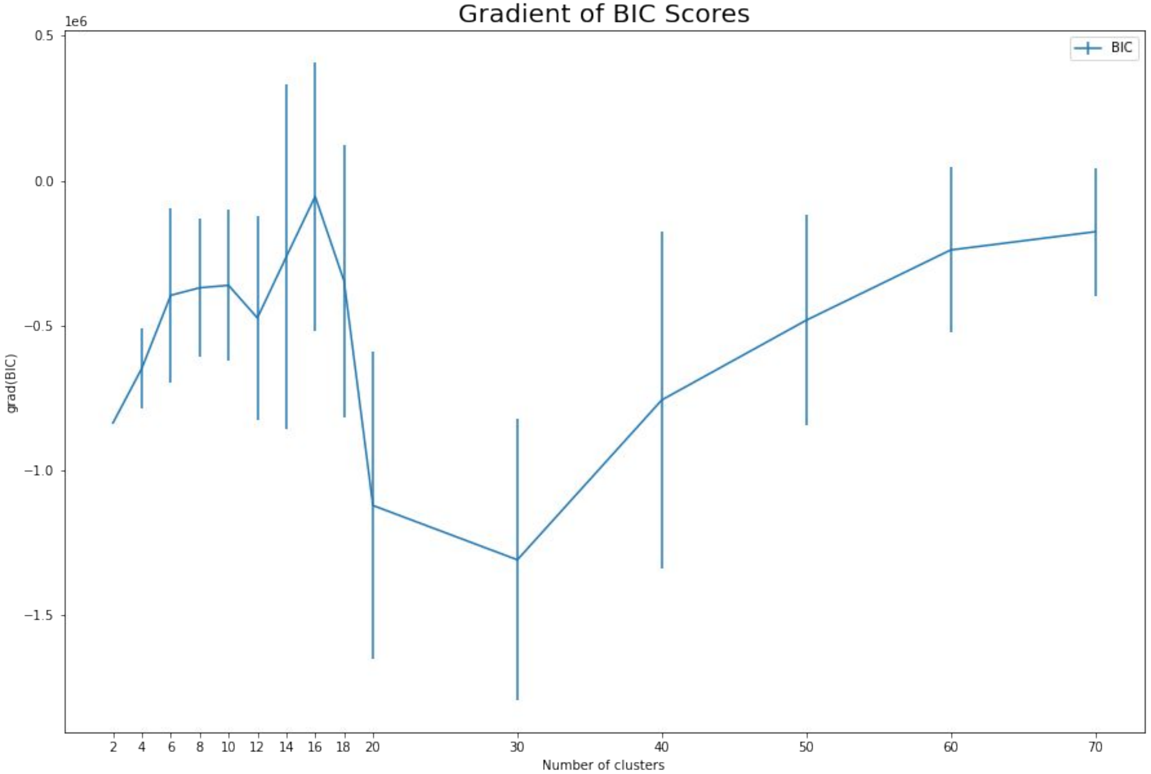
2. Some more GMM training

- n_components range: 2, 4, 6, …, 18, 20, 30, …, 70
- Fit models 10 times with same configuration per component
- Select the best 5 out of 10

➔ BIC score



```
for n_component in n_components_range:
 tmp_bic = []
 for iteration in range(iterations):

   gmm = GaussianMixture(n_components=n_component,
                         covariance_type='diag',
                         init_params='kmeans',
                         n_init=2).fit(X)
   tmp_bic.append(gmm.bic(X))
 val = np.mean(SelBest(np.array(tmp_bic), int(iterations/2)))
 err = np.std(tmp_bic)
 bics.apped(val)
 bics_err.append(err)
```



3. Jensen Shannon Distance

- Compute text probabilities from word probabilities
- JSDistance = sqrt(JSDivergence)

$$\mathrm{JSD}(P \parallel Q) = \frac{1}{2}D(P \parallel M) + \frac{1}{2}D(Q \parallel M)$$

Select:

- ➢ 1 article from Quora dataset
- ➢ 20 answering user comments
- ➢ 20 random user comments

		Diag - 20	**Diag - 70**
Answering User	mean	0.300	0.445
	stddev	0.075	0.079
Random User	mean	0.327	0.469
	stddev	0.056	0.057

4. Next Tasks

- Start experiments on different sets of datasets

References

- Pennington, Jeffrey, Richard Socher, and Christopher D. Manning. "Glove: Global vectors for word representation." *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 2014.
- Sridhar, Vivek Kumar Rangarajan. "Unsupervised topic modeling for short texts using distributed representations of words." *Proceedings of the 1st workshop on vector space modeling for natural language processing*. 2015.
- Fine tune GloVe embeddings using Mittens
- roamanalytics/mittens: A fast implementation of GloVe, with optional retrofitting
- https://medium.com/analytics-vidhya/basics-of-using-pre-trained-glove-vectors-in-python-d38905f356db
- https://scikit-learn.org/stable/modules/mixture.html
- https://towardsdatascience.com/gaussian-mixture-model-clusterization-how-to-select-the-number-of-components-clusters-553bef45f6e4
- https://stackoverflow.com/questions/26079881/kl-divergence-of-two-gmms
- https://medium.com/@sourcedexter/how-to-find-the-similarity-between-two-probability-distributions-using-python-a7546e90a08d

Questions

1. Different word embeddings - BERT?
2. Sentence embeddings?
3. Analysis based on single users or based on social medias?