Large-Scale Data Mining: Models and Algorithms **ECE 219 Winter 2018**
Prof. Vwani Roychowdhury TA: Qiujing Lu, Arash Vahabpour
UCLA, Department of ECE Tianyi Wang

Project 2: Clustering

DUE Monday Feb. 12, 2018 by 11:59 pm

Introduction

Clustering algorithms are unsupervised methods for finding groups of data points that have similar representations in a proper space. Clustering differs from classification in that no *a priori* labeling (grouping) of the data points is available.

K-means clustering is a simple and popular clustering algorithm. Given a set of data points $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ in multidimensional space, it tries to find K clusters s.t. each data point belongs to one and only one cluster, and the sum of the squares of the distances between each data point and the center of the cluster it belongs to is minimized. If we define $\boldsymbol{\mu}_k$ to be the "center" of the kth cluster, and

$$r_{nk} = \begin{cases} 1, & \text{if } \mathbf{x}_n \text{ is assigned to cluster } k \\ 0, & \text{otherwise} \end{cases}, \quad n = 1, \ldots, N \quad k = 1, \ldots, K$$

Then our goal is to find r_{nk}'s and $\boldsymbol{\mu}_k$'s that minimize $J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \|\mathbf{x}_n - \boldsymbol{\mu}_n\|^2$. The approach of K-means algorithm is to repeatedly perform the following two steps until convergence:

1. (Re)assign each data point to the cluster whose center is nearest to the data point.
2. (Re)calculate the position of the centers of the clusters: setting the center of the cluster to the mean of the data points that are currently within the cluster.

The center positions may be initialized randomly.

In this project, the goal includes:

1. To find proper representations of the data, s.t. the clustering is efficient and gives out reasonable results.
2. To perform K-means clustering on the dataset, and evaluate the performance of the clustering.
3. To try different preprocess methods which may increase the performance of the clustering.

Dataset

We work with "20 Newsgroups" dataset that we already explored in **Project 1**. It is a collection of approximately 20,000 documents, partitioned (nearly) evenly across 20 different newsgroups, each corresponding to a different topic. Each topic can be viewed as a "class".

In order to define the clustering task, we pretend as if the class labels are not available and aim to find groupings of the documents, where documents in each group are more similar to each other than to those in other groups. These groups, or clusters, capture the dependencies among the documents that are known through class labels. We then use class labels as the ground truth to evaluate the performance of the clustering task.

To get started with a simple clustering task, we work with a well separable portion of data that we used in Project 1, and see if we can retrieve the known classes. Namely, let us take all the documents in the following classes:

Table 1: Two well-separated classes

Class 1	`comp.graphics`	`comp.os.ms-windows.misc`	`comp.sys.ibm.pc.hardware`	`comp.sys.mac.hardware`
Class 2	`rec.autos`	`rec.motorcycles`	`rec.sport.baseball`	`rec.sport.hockey`

We would like to evaluate how purely the *a priori* known classes can be reconstructed through clustering.

Problem Statement

1. Building the TF-IDF matrix.

   Finding a good representation of the data is fundamental to the task of clustering. Following the steps in Project 1, **transform the documents into TF-IDF vectors**.

   Use `min_df = 3`, exclude the stopwords (no need to do stemming).

   Report the dimensions of the TF-IDF matrix you get.

2. Apply K-means clustering with $k = 2$ using the TF-IDF data. Compare the clustering results with the known class labels. (you can refer to sklearn - Clustering text documents using k-means for a basic work flow)

   (a) Inspect the contingency matrix to get a sense of your clustering result. Concretely, let **A** be the contingency table produced by the clustering algorithm representing the clustering solution, then A_{ij} is the number of data points that are members of class c_i and elements of cluster k_j.

   (b) In order to make a concrete comparison of different clustering results, there are various measures of purity for a given partition of the data points with respect to the ground truth. The measures we examine in this project are the **homogeneity score**, the **completeness score**, the **V-measure**, the **adjusted Rand score** and the **adjusted mutual info score**, all of which can be calculated by the corresponding functions provided in `sklearn.metrics`.

- Homogeneity is a measure of how "pure" the clusters are. If each cluster contains only data points from a single class, the homogeneity is satisfied.
- On the other hand, a clustering result satisfies completeness if all data points of a class are assigned to the same cluster. Both of these scores span between 0 and 1; where 1 stands for perfect clustering.
- The V-measure is then defined to be the harmonic average of homogeneity score and completeness score.
- The adjusted Rand Index is similar to accuracy measure, which computes similarity between the clustering labels and ground truth labels. This method counts all pairs of points that both fall either in the same cluster and the same class or in different clusters and different classes.
- Finally, the adjusted mutual information score measures the mutual information between the cluster label distribution and the ground truth label distributions.

Report the 5 measures above for the K-means clustering results you get.

3. Preprocess the data.

   As you may have observed, high dimensional sparse TF-IDF vectors do not yield a good clustering result. One of the reasons is that in a high-dimensional space, the Euclidean distance is not a good metric anymore, in the sense that the distances between data points tends to be almost the same (see [1]).

   K-means clustering has other limitations. Since its objective is to minimize the sum of within-cluster l^2 distances, it implicitly assumes that the clusters are isotropically shaped, *i.e.* round-shaped. When the clusters are not round-shaped, K-means may fail to identify the clusters properly. Even when the clusters are round, K-means algorithm may also fail when the clusters have unequal variances. A direct visualization for these problems can be found at sklearn - Demonstration of k-means assumptions.

   In this part we try to find a "better" representation tailored to the way that K-means clustering algorithm works, by preprocessing our data before clustering.

   (a) Dimensionality reduction

      i. We will use Latent Semantic Indexing (LSI) and Non-negative Matrix Factorization (NMF) that you are already familiar with for dimensionality reduction.
      First we want to find the effective dimension of the data through inspection of the top singular values of the TF-IDF matrix and see how many of them are significant in reconstructing the matrix with the truncated SVD representation. A guideline is to see what ratio of the variance of the original data is retained after the dimensionality reduction. **Report the plot of the percent of variance the top r principle components can retain v.s. r, for $r = 1$ to 1000.**

      ii. Now, use the following two methods to reduce the dimension of the data. Sweep over the dimension parameters for each method, and choose one that yields better results in terms of clustering purity metrics.

- Truncated SVD (LSI) / PCA
  Note that you don't need to perform SVD multiple times: performing SVD with $r = 1000$ gives you the data projected on all the top 1000 principle components, so for smaller r's, you just need to exclude the least important features.
- NMF

Specifically, try $r = 1, 2, 3, 5, 10, 20, 50, 100, 300$, and plot the 5 measure scores v.s. r for both SVD and NMF; also report the contingency matrices for each r.
Report the best r choice for SVD and NMF respectively.

How do you explain the non-monotonic behavior of the measures as r increases?

4. (a) **Visualize the performance of the case with best clustering results in the previous part your clustering by projecting final data vectors onto 2 dimensional plane and color-coding the classes.**

   (b) Now try the three methods below to see whether they increase the clustering performance. Still use the best r we had in previous parts.
   Visualize the transformed data as in part (a). Report the new clustering measures including the contingency matrix after transformation.
   - Normalizing features s.t. each feature has unit variance, i.e. each column of the reduced-dimensional data matrix has unit variance (if we use the convention that rows correspond to documents).
   - Applying a non-linear transformation to the data vectors only after NMF. Here we use logarithm transformation as an example.
     Can you justify why logarithm transformation may increase the clustering results?
   - Now try combining both transformations (in different orders) on NMF-reduced data.

5. Expand Dataset into 20 categories

   In this part we want to examine how purely we can retrieve all 20 original sub-class labels with clustering. Therefore, we need to include all the documents and the corresponding terms in the data matrix and find proper representation through dimensionality reduction of the TF-IDF representation. Still use the same parameters as in part 1.

   Try different dimensions for both truncated SVD and NMF dimensionality reduction techniques and the different transformations of the obtained feature vectors as outlined above.

Submission: Please submit a zip file containing your **report**, and your **codes** with a **readme file** on how to run your code to ee219.winter2018@gmail.com. The zip file should be named as "Project2_UID1_UID2_..._UIDn.zip" where UIDx's are student ID numbers of the team members. If you had any questions you can send an email to the same address.

References

[1] Why is Euclidean distance not a good metric in high dimensions? [online]. (https://stats.stackexchange.com/questions/99171/why-is-euclidean-distance-not-a-good-metric-in-high-dimensions).