Project 2: Clustering

Due Feb. 5, 2020 by 11:59 pm

Introduction

Clustering algorithms are unsupervised methods for finding groups of data points that have similar representations in a feature 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 such that each data point belongs to exactly one cluster, and that 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 $J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2$. 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 result of the clustering.
3. To try different preprocessing 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 category (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. 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 the data set that we used in Project 1, and see if we can retrieve the known classes. Specifically, let us define two classes comprising of the following categories.

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. That is, we take all the documents belonging to these two classes and perform unsupervised clustering into two clusters. Then we determine how pure each cluster is when we look at the labels of the documents belonging to each cluster.

Part 1 - Clustering of Text Data

1. Building the TF-IDF matrix.

   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 or lemmatization).

   QUESTION 1: Report the dimensions of the TF-IDF matrix you get.

2. Apply K-means clustering with $k = 2$ using the TF-IDF data. Note that the `KMeans` class in `sklearn` has parameters named `random_state`, `max_iter` and `n_init`. Please use `random_state=0`, `max_iter` ≥ 1000 and `n_init` ≥ 30[1]. 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) Given the clustering result and ground truth labels, contingency table **A** is the matrix whose entries A_{ij} is the number of data points that belong to both the class C_i the cluster K_j.

   QUESTION 2: Report the contingency table of your clustering result.

   (b) In order to evaluate clustering results, there are various measures for a given partition of the data points with respect to the ground truth. We will use the measures **homogeneity score**, **completeness score**, **V-measure**, **adjusted Rand score** and **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.

[1] If you have enough computation power, the larger the better

- 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.

QUESTION 3: Report the 5 measures above for the K-means clustering results you get.

3. Dimensionality reduction

   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 reducing the dimension of our data before clustering.

   We will use Singular Value Decomposition (SVD) and Non-negative Matrix Factorization (NMF) that you are already familiar with for dimensionality reduction.

   (a) 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.

   QUESTION 4: Report the plot of the percent of variance the top r principle components can retain v.s. r, for $r = 1$ to 1000.

   Hint: `explained_variance_ratio_` of `TruncatedSVD` objects. See sklearn document.

   (b) 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 / 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

QUESTION 5:
Let r be the dimension that we want to reduce the data to (*i.e.* `n_components`).
Try $r = \mathbf{1}, \mathbf{2}, \mathbf{3}, \mathbf{5}, \mathbf{10}, \mathbf{20}, \mathbf{50}, \mathbf{100}, \mathbf{300}$, and plot the 5 measure scores v.s. r for both SVD and NMF.
Report a good choice of r for SVD and NMF respectively.

Note: In the choice of r, there is a trade-off between the information preservation, and better performance of k-means in lower dimensions.

QUESTION 6: How do you explain the non-monotonic behavior of the measures as r increases?

4. (a) Visualization.
We can visualize the clustering results by projecting the dim-reduced data points onto 2-D plane with SVD, and coloring the points according to

- Ground truth class label
- Clustering label

respectively.

QUESTION 7: Visualize the clustering results for:

- SVD with your choice of r
- NMF with your choice of r

(b) Now try the transformation methods below to see whether they increase the clustering performance. Perform transformation on SVD-reduced data and NMF-reduced data, respectively. Still use the best r we had in previous parts.

- Scaling 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 logarithmic non-linear transformation to the data vectors for the case with NMF (non-negative features):

$$f(x) = \log(x + \epsilon), \quad 0 < \epsilon \ll 1$$

- Try combining the above transformations for the NMF case.

To sum up, try the SVD case w/ and w/o performing scaling (2 possibilities). Similarly, try different combinations of w/ and w/o performing scaling and non-linarity for the NMF case (4 possibilities).

QUESTION 8: Visualize the **transformed** data as in part (a).

QUESTION 9: Can you justify why the "logarithm transformation" may improve the clustering results?

QUESTION 10: Report the clustering measures (except for the contingency matrix) for the transformed data.

Part 2 - Your Own Dataset

Be creative and choose an interesting dataset of your own, from your research or elsewhere. Inspired by part 1, perform clustering analysis.

- Report your pipeline and explain how you extracted meaningful features from your data, how well the clustering performed, etc.
- Make sure your dataset is not too trivial for a learning algorithm. Moreover, in this part, your data should include more than two classes, e.g. something between 5 and 10.

Part 3 - Color Clustering

In this part we would like to perform "segmentation" on images based on color clustering. Choose an image of size $m \times n$ of a celebrity's face. Reshape your image to a matrix of size $mn \times 3$, where the size 3 stems from the RGB channels. Transform pixel RGB information to "normalized (r, g) space" where:

$$r = \frac{R}{R+G+B}, \qquad g = \frac{G}{R+G+B}.$$

Choose a small number of clusters, cluster the pixels according to their colors, and report your result. Here is a sample result:

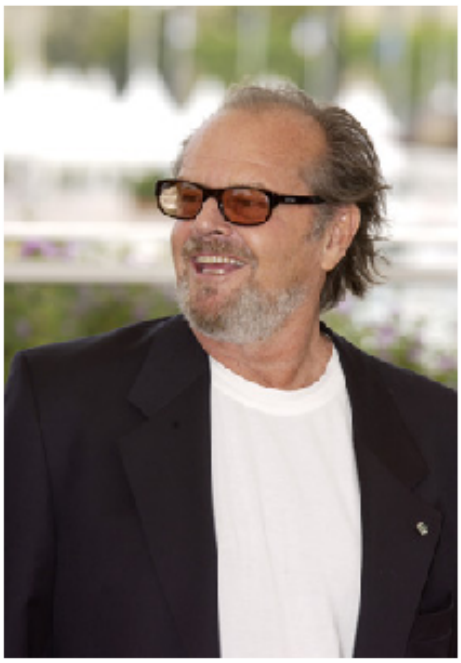

Figure 1: Original figure

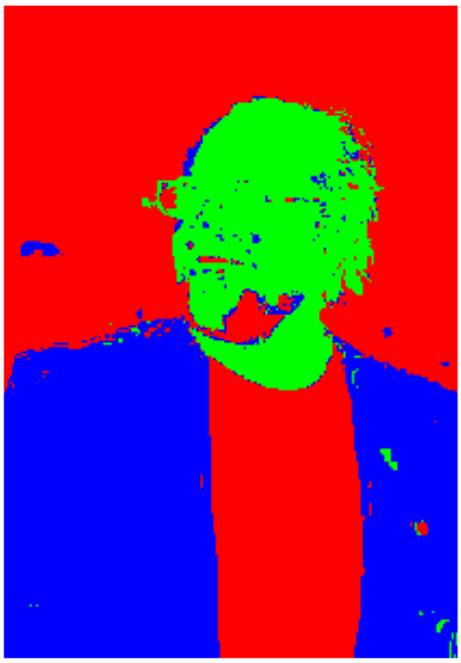

Figure 2: Result of k-means clustering

QUESTION 11: BONUS - can you suggest a methodology to make an appropriate choice of k and initial seeds of cluster centers?

Submission

Please submit a zip file containing your **report**, and your **codes** with a **readme file** on how to run your code to CCLE. The zip file should be named as "Project2_UID1_UID2_..._UIDn.zip" where UIDx's are student ID numbers of the team members.

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).