Abstract: This report talks about two popular clustering approaches: *k-means clustering and spectral clustering*. It first presents the flow of both algorithms. Then it discusses their performance based on external and internal index evaluation. Lastly, it compares the pros and cons of these two clustering algorithms.

The Flow of Two Clustering Algorithms

Data processing:
For both clustering algorithms, there are some common steps, namely the data preprocessing part. After loading the files, we observe that outliers exist and some columns are not part of the attributes that we are classifying.

- Remove the rows/data points that have values of -1
- Remove the first two columns of each data set
- Normalize entire data sets via Min-Max method

K-means:
I implemented my own k-means algorithm (without calling the built-in package), which could be slightly different from the more general ones. For example, the initialization strategy could be varied rather than a randomized selection. In addition, the stopping criterion could be that the centroids stay unchanged instead of a fixed number of iterations.

- Randomly select k number of data points to be the initial centroids
- Assign each data point to its closest centroid
- Calculate the mean of each cluster, which becomes the new centroid
- Update the centroids and re-assign each data point to its closest centroid
- Repeat the previous two steps for 1000 iterations

Spectral Clustering:
I called the built-in spectral clustering function with a k-nearest neighbor graph to classify each data point to its cluster. The general flow of spectral clustering is listed below.

- Create a similarity graph in the form of an adjacent matrix
    - ε-neighborhood graph
    - k-nearest neighbor graph
    - fully-connected graph
- Project the data onto a lower-dimensional space by calculating Graph Laplacian Matrix
- Run Kmeans to cluster data points

Performance Evaluation

In order to evaluate the performance of different clustering algorithms, external and internal indexes are needed. I choose the sum of squares as my internal index and purity as my external index. The formulas are as follow:

Sum of squares(internal index)	Purity(external index)
$\mathrm{SSE} = \sum_{\underbrace{i=1}_{\text{test set}}}^{n} (\underbrace{y_i}_{\text{predicted vaue}} - \underbrace{\hat{y}_i}_{\text{actual value}})^2$	$\mathrm{purity}(\Omega, \mathbb{C}) = \frac{1}{N}\sum_k \max_j \lvert\omega_k \cap c_j\rvert$

Below shows the results:

```
The sume of square errors for each cluster in cho are
cluster[ 1 ]:  38.20745258213443
cluster[ 2 ]:  35.61380665974774
cluster[ 3 ]:  45.66905767874218
cluster[ 4 ]:  47.70528417979411
cluster[ 5 ]:  58.27888470410556
The purity for cho with kmeans is  0.6870562681850636
The sume of square errors for each cluster in iyer are
cluster[ 1 ]:  56.912077420698104
cluster[ 2 ]:  5.258815370431632
cluster[ 3 ]:  5.937196366600489
cluster[ 4 ]:  12.57137496865832
cluster[ 5 ]:  38.621014540053636
cluster[ 6 ]:  13.841146759910021
cluster[ 7 ]:  11.51933421023877
cluster[ 8 ]:  4.746533132189539
cluster[ 9 ]:  5.069219497887325
cluster[ 10 ]:  16.192114321544512
The purity for iyer with kemeans is  0.6889346176613178
```

results of running K-means algorithm on two data sets: Cho & Iyer

```
The sume of square errors for each cluster in cho are
cluster[ 1 ]:  37.271510013085646
cluster[ 2 ]:  41.06730444513531
cluster[ 3 ]:  76.87462354352782
cluster[ 4 ]:  19.39525290705363
cluster[ 5 ]:  56.35864362334264
The purity for cho with spectral clustering is  0.620431666055684
The sume of square errors for each cluster in iyer are
cluster[ 1 ]:  39.60205227467691
cluster[ 2 ]:  1.2498067718492183
cluster[ 3 ]:  14.158167990060816
cluster[ 4 ]:  9.08225728335192
cluster[ 5 ]:  7.049835183870017
cluster[ 6 ]:  13.956227611115663
cluster[ 7 ]:  15.401683516709614
cluster[ 8 ]:  27.836066642486518
cluster[ 9 ]:  27.441152307317427
cluster[ 10 ]:  7.860131802951841
The purity for iyer with spectral clustering is  0.7479646935665317
```

results of running spectral clustering algorithm on two data sets: Cho & Iyer

From the results, we can see that the purity for both algorithms is roughly around 0.7. The sum of squares has a similar distribution as well.

Comparison Between Two Algorithms

Pros of K-means:

- It is very simple to understand and easy to implement.
- It is efficient with time complexity of O(tkn), where t is the number of iterations, k is the number of clusters, and n is the number of data points.
- It's suitable for large data sets.

Cons of K-means:

- The algorithm is only applicable if the mean is defined.
- Programmers have to specify the number of clusters.
- The choice of initial points can have a big impact on the results.
- It only works well when the shape of the clusters is hyper-spherical.
- With a random choice of centroids, it may yield different clustering results on different runs of the algorithm. Thus, the results may not be repeatable and lack consistency.
- It is sensitive to outliers.
- It does not work well with clusters of a different size or density

Pros of spectral clustering:

- It is more general because whenever K-means is appropriate for use then so too is spectral clustering. The reverse is not true.
- No issues of getting stuck in local minima or restarting the algorithm several times with different initializations

Cons of spectral clustering:

- It is more complex because it needs a few more steps before applying K-means such as constructing a similarity graph.
- It is inefficient with time complexity of O(n^3).
- It is computationally expensive.

Conclusion

Based on the experiment of K-means and spectral clustering, there is no clear evidence that any other clustering algorithm performs better in general, although they may be more suitable for some specific types of data or applications. In addition, comparing different clustering algorithms is a difficult task, because no one knows exactly what the correct clusters are.