Comparison of Clustering Methods using YELP dataset

Angadpreet Nagpal
University of Southern California
asnagpal@usc.edu

Azamat Ordabekov
University of Southern California
ordabeko@usc.edu

ABSTRACT

This paper presents the experimental study on the common Clustering methods - K-means, Bisecting K-means, Spectral Clustering and Gaussian Mixture Model. These comparisons are based on both visual and empirical study done on the YELP dataset. Further evaluation is done on the clustering algorithms by changing the number of clusters, attributes, and the dataset.

KEYWORDS

Clustering, Spark, KMeans, Bisecting KMeans, Spectral Clustering, YELP, GMM

1 INTRODUCTION

Clustering is an unsupervised learning problem for examining the "points" and grouping them into "clusters" according to some distance measure. The goal is to bring similar items together in one cluster while dissimilar items belong to other cluster. We implemented clustering using four algorithms - K Means, Bisecting K Means, Gaussian Mixture Model and Spectral/Power Iteration Clustering. Each of these methods take a different approach to clustering data and will be examined in the paper below.

The Yelp dataset is a subset of businesses, reviews, and user data for use in personal, educational, and academic purposes. The challenge is to use the data in innovative ways and break ground in research. The data used in this analysis is user dataset and business dataset.

Initial comparisons are done on the user dataset. For reproducability of results, the baseline for user dataset is set at 20 iterations, 4 clusters, random initial initialization and seed 2018. The comparison is done using Silhouette Value comparison. The distance measure used here is Euclidean distance.

2 DATASET

2.1 User Dataset

User data includes the user's friend mapping and all the metadata associated with the user. The user data is created using the following attributes - yelping_since (when the user joined Yelp, in terms of days ranging from 354 to 4293 days), average_stars (average rating of all reviews ranging from 1 to 5), review_count (the number of reviews they've written ranging from 1 to 179). The values are scaled using MinMaxScaler to make the values between 0 and 1. Finally this dataset is input into the algorithms.

User data set does not meet the underlying data assumptions for any of the algorithms. It is not spherical, uniform, or normal and doesn't have a dominant Eigen vector. This data set is hence non ideal for all algorithms. This will lead to a good comparison and hence is used for the first part of the analysis.

2.2 Business Dataset

Business data contains business data including location data, attributes, and categories. It is used in the later part of the evaluation and is prepared in the same way as before but with stars(star rating, rounded to half-stars ranging from 1 to 5), is_open(0 or 1 for closed or open, respectively), review_count(number of reviews ranging from 3 to 1565) attributes.

The business dataset is highly clusterable, conforming to the underlying assumptions for all the algorithms. It is spherical, uniform, normal and contains a dominant eigen vector. Hence this will give a very good result for all the algorithms.

3 COMPARISON MEASURE

Silhouette value for an algorithm refers to a method of interpretation and validation of consistency within clusters of data. The technique provides a succinct graphical representation of how well each object lies within its cluster.

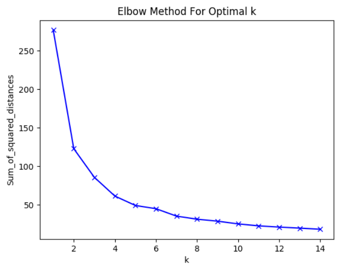
The silhouette value is a measure of how similar an object is to its own cluster (cohesion) compared to other clusters (separation). The silhouette ranges from −1 to +1, where a high value indicates that the object is well matched to its own cluster and poorly matched to neighboring clusters. If most objects have a high value, then the clustering configuration is appropriate. If many points have a low or negative value, then the clustering configuration may have too many or too few clusters.

For each datum i, let a(i) be the average distance between i and all other data within the same cluster. We then define the average dissimilarity of point i to a cluster c as the average of the distance from i to all points in c. Let b(i) be the smallest average distance of i to all points in any other cluster, of which i is not a member. We now define a silhouette:

$$s(i) = \frac{b(i) - a(i)}{max(b(i), a(i))}$$

4 CHOOSING APPROPRIATE K

The elbow method looks at the percentage of variance explained as a function of the number of clusters: One should choose a number of clusters so that adding another cluster doesn't give much better modeling of the data.



The K chosen in this case is 4.

5 CLUSTERING ALGORITHMS

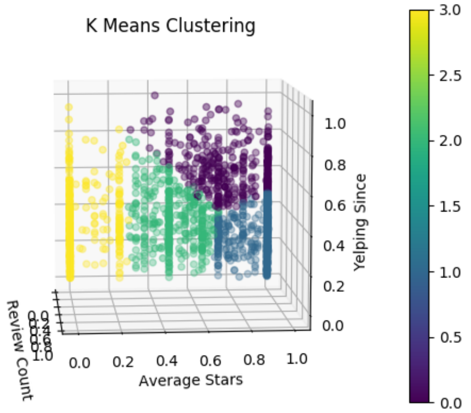
5.1 K Means

K-Means clustering intends to partition n objects into k clusters in which each object belongs to the cluster with the nearest mean. This method produces exactly k different clusters of greatest possible distinction. It is an unsupervised iterative algorithm that groups input data in a predefined number of k clusters. Each cluster has a centroid which is a cluster center.

The algorithm selects k points at random as cluster centers. It assigns objects to their closest cluster center according to the Euclidean distance function. Calculate the centroid or mean of all objects in each cluster. Repeat until the same points are assigned to each cluster in consecutive rounds.

KMeans class is an implementation of the K-means clustering algorithm in machine learning with support for k-means parallel in Spark MLlib.

Following is a visualization of clusters with K Means Clustering -



The Silhouette value for the algorithm is 0.583137255226948
The number of elements in each clusters are -

Cluster Number	Number of Elements	Per Cluster Silhouette Value
0	365	0.3609812916531975
1	591	0.7274430110197224
2	402	0.526547759149968
3	342	0.6363682744723312

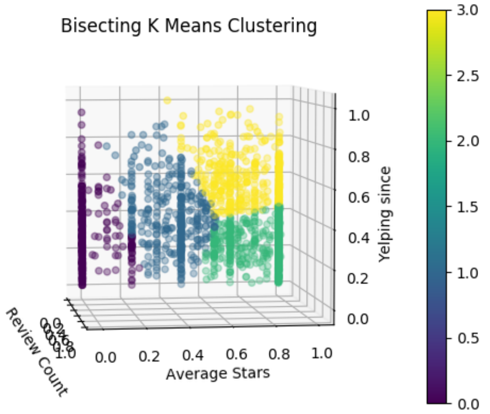
Interpretation of the clusters Cluster 0 (Purple data points) represents the most valuable customers who have been yelping for a long time, have given good average stars and have a high review count. On the other hand, Cluster 3 (Yellow) represents the users who give very low average stars, don't have a high review count and may or may not have been yelping for a long time. Cluster 1 (Blue) and 2 (Green) are normal users.

5.2 Bisecting K Means

The bisecting K-means is a divisive hierarchical clustering algorithm and is a variation of K-means. Similar to K-means, the number of clusters must be predefined. Spark MLlib also provides an implementation for bisecting K-means algorithm.

The algorithm starts from a single cluster that contains all points. Iteratively it finds divisible clusters on the bottom level by finding the most dissimilar pair of clusters in the current cluster. It then bisects each of them using k-means, until there are k leaf clusters in total or no leaf clusters are divisible. The bisecting steps of clusters on the same level are grouped together to increase parallelism. If bisecting all divisible clusters on the bottom level would result more than k leaf clusters, larger clusters get higher priority.[1]

Following is a visualization of clusters with Bisecting K Means -



The Silhouette value for the algorithm is 0.5675392294770845
The number of elements in each clusters are -

Cluster Number	Number of Elements	Per Cluster Silhouette Value
0	296	0.6790278425662956
1	385	0.4654577037767246
2	631	0.6843144809683361
3	388	0.39386802903701135

Interpretation of the clusters Same as K Means.

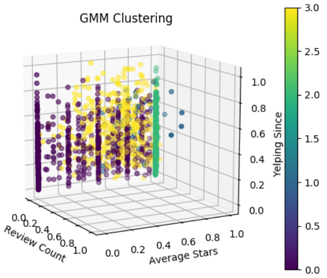
5.3 Gaussian Mixture Model (GMM)

A Gaussian Mixture Model represents a composite distribution whereby points are drawn from one of k Gaussian sub-distributions, each with its own probability. The spark.ml implementation uses the expectation-maximization algorithm to induce the maximum-likelihood model given a set of samples.

This class performs expectation maximization for multivariate Gaussian Mixture Models (GMMs). A GMM represents a composite distribution of independent Gaussian distributions with associated "mixing" weights specifying each's contribution to the composite.

Given a set of sample points, this class will maximize the log-likelihood for a mixture of k Gaussians, iterating until the log-likelihood changes by less than convergenceTol, or until it has reached the max number of iterations. While this process is generally guaranteed to converge, it is not guaranteed to find a global optimum.

Following is a visualization of clusters with GMM -



The Silhouette value for the algorithm is 0.12813702784322586 The number of elements in each clusters are -

Cluster Number	Number of Elements	Per Cluster Silhouette Value
0	785	-0.14262559342207642
1	35	0.04844638201936253
2	503	0.6509330427731185
3	377	0.001801311098826752

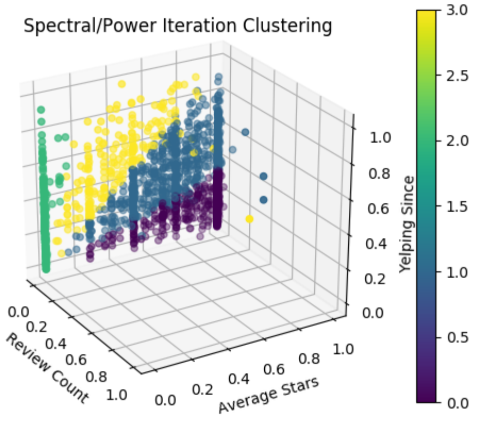
Interpretation of the clusters Cluster 1 (Blue data points) represents the users who have review count greater than 0.2. Cluster 3 (Yellow) represents all the users having review count greater than 0.0 and less than 0.2. Cluster 0 and Cluster 2 are mixed together and have review count equal to 0.0

5.4 Power Iteration Clustering (PIC)

Power Iteration Clustering (PIC), a scalable graph clustering algorithm developed by Lin and Cohen. From the abstract: PIC finds a very low-dimensional embedding of a dataset using truncated power iteration on a normalized pair-wise similarity matrix of the data.

Start by creating a similarity graph between our N objects to cluster. Compute the first k eigenvectors of its Laplacian matrix to define a feature vector for each object. Run k-means on these features to separate objects into k classes.

Following is a visualization of clusters with PIC -



The Silhouette value for the algorithm is 0.3353219128346997

The number of elements in each clusters are -

Cluster Number	Number of Elements	Per Cluster Silhouette Value
0	550	0.5966287188010183
1	707	0.05539239260562336
2	235	0.767223538290254
3	208	0.10789184330790988

Interpretation of the clusters Cluster 2 (Green data points) represents the users who give the least average stars, least review count and according to the algorithm are the most similar to each other. These points represents the least useful users for business. Cluster 1 (Blue) contains the most number of points and has the least silhouette value. Cluster 0 (Purple) are quite similar to each other and contains the second most number of points.

6 INITIAL COMPARISON AND CONCLUSIONS

The initial comparison based on the baseline measure(user dataset, 4 clusters, random initialization, 20 iterations) gives us the following observations -

- K Means performs the best on the YELP user dataset.
- Bisecting K Means gives performance comparable to K Means.
- PIC gives good clusters with 0.33 Silhouette value.
- GMM performs the worst with 0.13 Silhouette value.
- PIC gives the best quality cluster with Cluster 2 having Silhouette value 0.77

Why K Means seems to perform the best?

The dataset plays an important role in this. Even though the dataset has been normalized using MinMaxScaler. GMM assumes the dataset to have been drawn from k Gaussian distributions which is not true for our dataset. Similarly, PIC which works by computing the similarity graph and computing the eigen vectors expects the existence of a dominant eigen value, which again doesn't exist in our dataset. Finally, Bisecting K Means gives a similar result to K Means which is still quite low at 0.58 due to the fact that K Means also assumes that the dataset be spherical which is not true in our case. Bisecting K Means is a variation of K Means and is always expected to work better than K Means or sometimes just slightly worse than K Means.

7 COMPARISON BY NUMBER OF CLUSTERS

Experimenting with the number of clusters for each algorithm while keeping all the other parameters the same, we get the following table

Number of Clusters	KMeans	Bisecting KMeans	GMM	PIC
3	0.58	0.57	-0.09	0.38
4	0.58	0.56	0.13	0.34
5	0.59	0.52	0.19	0.23

The Silhouette value shows a decreasing trend as the number of clusters increases for all the algorithms.

Why these results

If we look at the formula for silhouette value, s(i) = $\frac{b(i) - a(i)}{max(b(i), a(i))}$

where b(i) represents the dissimilarity in the cluster, and a(i) represents the similarity in the cluster. As the number of clusters increases, the dissimilarity within the clusters decreases while the similarity increases. Hence the numerator is decreasing and the denominator is also decreasing. This leads to the decrease in the silhouette value as the number of clusters increase.

There may be cases of increasing silhouette value with increase in number of clusters due to less relative decrease in numerator than the decrease in the denominator. However, the overall trend will always be decreasing silhouette value with increase in number of clusters.

8 EFFECT OF OUTLIERS

We get the following table by by keeping and removing the outliers from the dataset while keeping all the other parametrs the same,

	KMeans	Bisecting KMeans	GMM	PIC
Without Outliers	0.58	0.57	0.24	0.38
With Outliers	0.58	0.57	0.13	0.34

Below are the observations -

- The performance of GMM and PIC decreases by including the outliers.
- K Means and Bisecting K Means are not affected.

Why these results?

GMM is highly sensitive to outliers going from 0.24 to 0.13. This is because the algorithm works on the concept of Gaussaian distribution which is highly affected by outliers. PIC is sensitive to outliers but not as much as GMM. This is because outliers leads to decrease in the similarity.

9 COMPARISON BY ATTRIBUTES

The comparison by attributes gives information about how each attribute affects clusters and silhouette value itself. The experiment was provided by using K-means algorithm since it gives the best fit on user dataset.

Attributes	Result
average_stars, yelping_since, review_count	0.58
average_stars, yelping_since (Without review_count)	0.595
review_count, yelping_since (Without average_stars)	0.64
average_stars, review_count (Without yelping_since)	0.76

These gives us the following observations -

- "Review count" and "average stars" don't affect clustering that much with very less increase in silhouette score.
- "Yelping since" contributes the most to clustering with the most change in the silhouette score

Why these results

"Yelping since" is itself quite variant for each user. Hence, it provides the most variation to the user dataset in turn contributing towards the dissimilarity, whereas "Review count" and "average stars" don't show that much variation between users.

10 TIME COMPLEXITY

	K-means	Bisecting K-means	GMM	PIC
Time Complexity	O(IKN)	O((K-1)IN)	O(IKN)	O(IN^3)

where K - number of clusters. I - number of iterations. N - number of data points.

11 COMPARISON ON BUSINESS DATASET

Experimenting on business dataset (highly clusterable) for each algorithm with 3 clusters as calculated by the Elbow method, we get the following table

	KMeans	Bisecting KMeans	GMM	PIC
Silhouette value	0.78	0.73	0.72	0.73

These are the observations -

- Performance for K Means is the highest.
- Performance for Bisecting K Means, GMM and PIC are similar.

Why these results?

The dataset is ideal for clustering conforming to the underlying assumptions for all the algorithms and hence the silhouette value is similar for all the algorithms.

12 CONCLUSION

We get the following conclusions from the analysis -

(1) K Means operates under the assumption that the dataset should be spherical.
(2) GMM operates under the assumption that the dataset is taken from a Gaussian distribution.
(3) PIC operates under the assumption that the dataset should be similar and have a dominant eigen value.
(4) K Means performs the best on the user dataset with Bisecting K Means not being far behind.
(5) GMM is highly sensitive to outliers affecting the distribution attributes.
(6) PIC is also sensitive to outliers affcting the similarity between the cluster elements.
(7) Time complexity wise, PIC is the most computationally expensive.
(8) Attribute showing the most variation contribute the most towards clustering.
(9) If the dataset conforms to the underlying assumption of the algorithm, it will show very good result.

Overall, clustering hugely depends on the dataset properties. K-Means expects data to be spherical. Bisecting K-Means expects uniformly distributed data. GMM works good on Gaussian distribution and fails when outliers are present in the dataset. PIC works good on data which is similar to each other and has eigen vectors with dominant values.

13 REFERENCES

[1] Steinbach, M., Karypis, G., Kumar, V., "A Comparison of Document Clustering Techniques," University of Minnesota, Technical Report #00-034 (2000).
http://www.cs.umn.edu/tech_reports/

A CODEBASE

The code can be found at https://github.com/angadp/YELPClustering. Please refer to the commit history to find the contribution history.