Clustering

Bethany Ludwig

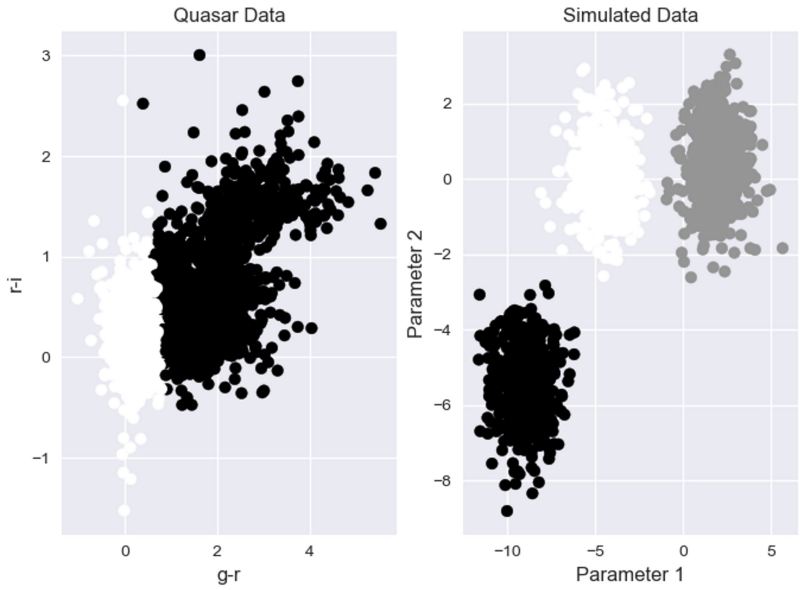
Friday, April 26, 2019 1:26 PM

Due by 2 PM on Monday April 29th

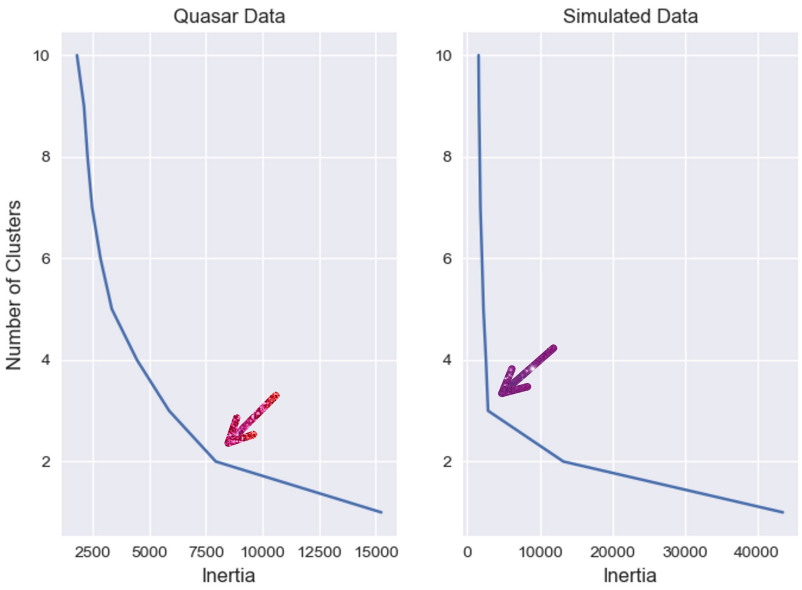
- You'll need two data sets with at least two features:

- One that has elongated clusters or irregular shapes in the distribution. It should be easy to find a real astrophysical data set like this.

- One that is convex and isotropic with blobs of approximately equal numbers of points. You'll probably have to make this yourself.



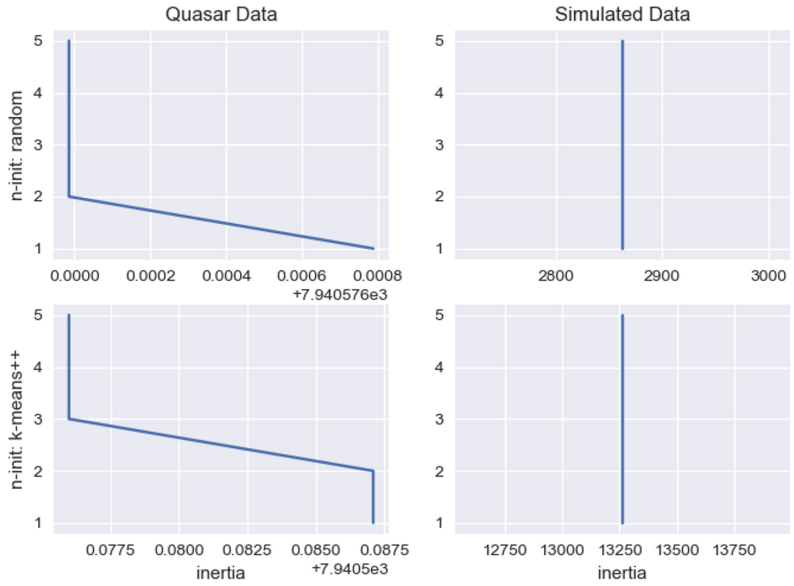
- Run the `kmeans` module on each data set with a range of different number of clusters. Plot the number of clusters vs. inertia for each. (You can do this easily using the `inertia_` attribute.)



- Choose an optimal number of clusters. Write a sentence justifying why. (e.g. Visual inspection? Cross validation using labels?)

For the quasar and simulated data I chose 2 and 3 clusters respectively. This is for two reasons. The first is that I know the quasar data is typically split up into low and high redshift and I specifically set the simulated data to have 3 clusters. The second way I evaluate the optimal number of clusters is using "the elbow method." The idea is that we want a number of clusters such that the inertia is minimized at a point before the returns you get from increasing the number of clusters is diminished. The "elbows" in these plots are consistent with the number of clusters I chose from visually looking at the data and prior knowledge.

- Using that number of clusters, run the `kmeans` module on each data set with a range of different `n_init` values. For each data set, plot `n_init` vs. inertia for both `init='random'` and `init='kmeans++'` .



- Discuss the trends you see. If you don't see any, try using manually placing the initial positions of the centroids at strange locations so they end up in local minima.

For the simulated data, running the kmeans module multiple times has no effect regardless of which initialization method is imposed. The data is separated cleanly without much overlap so I think that it makes sense that determining clusters would only require one step.

The quasar data, on the other hand, is effected but with a very small difference in inertia and even still only requires 2 or 3 runs before it reaches a minimum. What's strange is that k-means++ algorithm requires an additional step than the random algorithm despite being a more efficient process than just choosing centroids randomly. This could simply be a consequence of having only two clusters and an infinitesimal difference in inertia.

- Save all of this as a single PDF with your name somewhere.
Python modules you'll want to use:
sklearn.cluster.KMeans