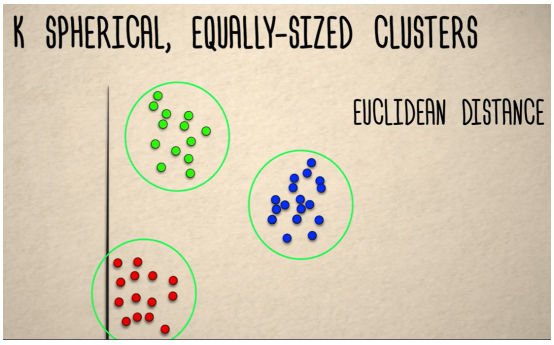
Beyond K-Means: Other Notions of Distance

K-Means clustering is a powerful framework for clustering. But we have seen some instances where K-Means clustering is not always the best clustering framework for a particular problem.

As we know clustering is about grouping data by similarity and in the last article we have briefly looked at how we might be interested in different notions of groupings or clustering beyond K-Means.



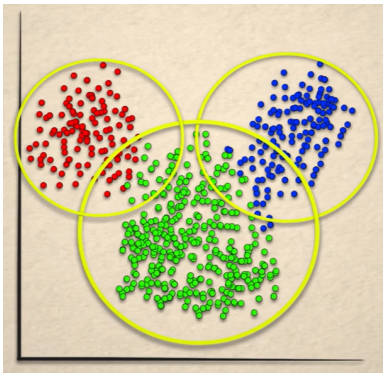
Here,we will discuss how some time we define **Similarity** differently than the definition used by K-Means clustering.

K-Means clustering assumes that we have K spherical equally sized clusters. This follows from the use of the Euclidean Distance as the dissimilarity measure for K-Means.

Using the Squared Euclidean Distance in our objective function is equivalent to using Euclidean Distance and does not change the fact.

So, what is the problem with the K-Means definition of similarity?

First consider the case where our clusters are not equally sized. For instance consider an example where we have small circles filled with data points adjacent to a very large circle filled with data points. We might naturally think there are 3 clusters corresponding to the 3 circles.



In the K-Means objective, the clustering where all the data points in a large circle are assigned to the same cluster has relatively higher or worse objective value, and the clustering where the data points in the large circle are closest to the small circles and are assigned to the outer clusters that have a relatively lower or better objective value.

This is because the data points at the edges of the large circle are relatively far from the cluster center of the large circle, but they are relatively close to the cluster centers in the small circles.

Now the question is, How can we really separate out the three circles as clusters?

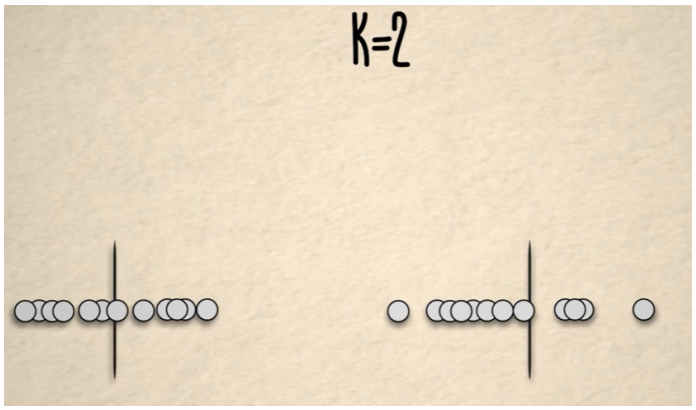
One option is to use a model that specifically encodes clusters of different sizes.
For instance, previously we mentioned Gaussian Mixture Models. When the mixture components are allowed to have different covariances, then these models can capture the kind of clusters that we are trying to find here.

One useful algorithm for this type of model is called the **Expectation Maximization or EM** algorithm.

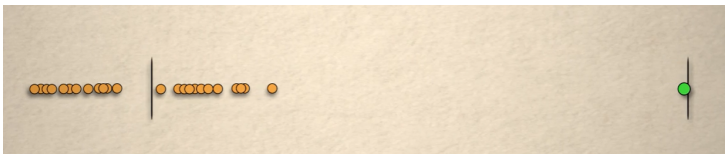
Another issue here is, we might encounter outliers in data. Outliers are data points that are relatively far away from most of the data points in the dataset..

As K-Means Clustering uses Euclidean Distance, that's why K-Means Clustering is typically very sensitive to outliers.

Consider an example of one dimensional dataset:
If we run the K-Means algorithm with K=2, we find that there are two clusters as we expect.



But if we add a single point very far away from the data, then we get one cluster with all the main dataset and the second cluster contains only the outlier point. This cluster does not really capture the two clusters that really exist in the data.



So, what can we do in such a situation?
One option is to use an alternative measure of distance, that is less sensitive to outliers.

Previously to calculate the Squared Euclidean Distance, we added up the squared difference between every pair of features shared by the two data points. This is also called the L2 distance.

$$dis(x_3, x_{17}) = \sum_{d=1}^{D} (x_{3,d} - x_{17,d})^2$$

L^2 DISTANCE

An alternative distance is the **L1 distance**, where we say the distance is the sum of the **absolute values** of the differences between every pair of features shared by two data points.

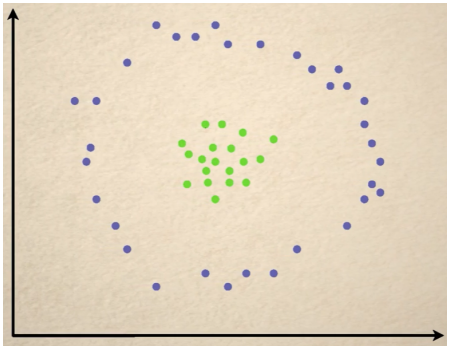
$$dis(x_3, x_{17}) = \sum_{d=1}^{D} |x_{3,d} - x_{17,d}|$$

L^1 DISTANCE

Pretty much everything we already did can be adapted to this case. In particular, we can also derive an iterative algorithm like K-Means, for this problem. The algorithm is called **K-Medoids**, since the optimal cluster centers are now medoids instead of means. The medoid in one dimension is the median.

Finally we might be interested in clusters of different shapes, rather than only spherical clusters.

Consider the example dataset, here we can see two identifiable groups or clusters of points. But one group is situated inside the other. If we apply K-Means with



two centers, then we would just split the data into two sides of some essentially straight lines.

We need something different to capture the two clusters that we can see here.

- For instance, we might define a radial notion of similarity.
- We can also consider splitting our data into polar coordinates or using kernel methods.
- Another alternative is to use Agglomerative Clustering.

Another potential issue with applying K-Means clustering is that it requires continuous numerical features. If we are dealing with text or binary YES/NO features that are not continuous numerical, then we might consider alternative notions of similarity.

Another option here could be to transform our data into a form, which is more amenable to K-Means Clustering.