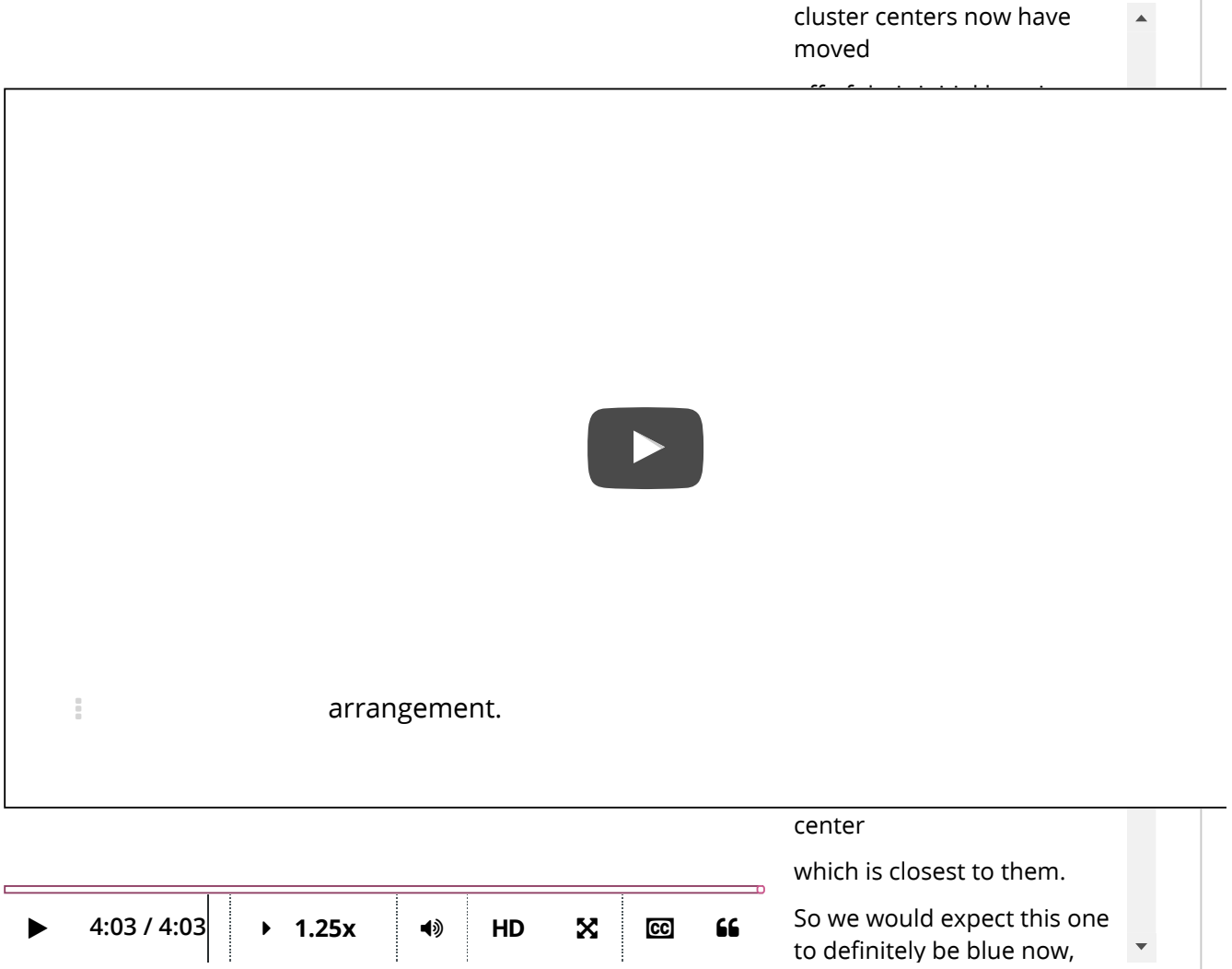
Video

Bookmark this page

How Does K-Means Work?



Video

Download video file

Transcripts

Download SubRip (.srt) file

Download Text (.txt) file

Clustering groups samples that are similar within the same cluster. The more similar the samples belonging to a cluster group are (and conversely, the more dissimilar samples in separate groups), the better the clustering algorithm has performed. Since clustering is an

unsupervised algorithm, this similarity metric must be measured automatically and based solely on your data.

The implementation details and definition of *similarity* are what differentiate the many clustering algorithms. The **K-Means** way of doing this, is to iteratively separate your samples into a user-specified number of "K" cluster groups of roughly equal variance. Cluster groups are defined by their geometric cluster center, single point referred to as its centroid. Separately, *centroid* and *cluster* are sometimes used interchangeably; but if used together, a cluster is a set of similar samples, and a centroid is just the mean feature-position of all samples assigned to the cluster.

The centroids are not records in your dataset, however they do 'exist' within your datasets feature-space. This is important because it allows for a meaningful distance measure to be calculated between the centroids and your samples. Every sample in your dataset is assigned to the centroid *nearest* to it, so if you have a sample that is 10 units away from CusterA's centroid, and 100 units away from ClusterB's, the sample is assigned to ClusterA.

In the case of continuous features, calculating the distance is straightforward. But when you have categorical features, such as 'Cookies n Cream' vs 'Mango' ice cream favors, you'll have to creatively come up with other methods. SciKit-Learn's K-Means implementation only natively supports numeric features types, so we'll leave the discussion on how to do clustering with categorical features to the Dive Deeper section.

The K-Means Algorithm

K-Means starts by placing a user-specified number of "K" cluster centers in your feature space. There are many techniques for choosing the first centroid placement, and your results will vary depending on the one you select! The simplest being just use the position of some random samples as the centroids' starting spots.

Each cluster then takes ownership of the samples nearest to its centroid, and every sample can only be assigned as single cluster. 'Nearest' is a value that has to be evaluated and in SciKit-Learn, it is defined as the multivariate, *n*-dimensional Euclidean distance between the sample and the centroid. After this, the centroid location is updated to be the mean value of all samples assigned to it. This mean value is calculated by feature, so the centroid position ends up being a n-length vector within your feature space.

The assignment and update steps repeat until there are no more changes in either, at which point the algorithm has converged. K-Means *always* converges, and it is very fast at doing so. But it does not always converge at the *global* minima...

The technical explanation for what K-Means does is minimizing the within-cluster inertia, or **sum of squared errors** between each sample and its respective centroid. As mentioned, the initial centroid assignment affects the results. Two runs of K-means might

produce different outcomes, but the *quality* of their cluster assignments are ranked by looking at which run has the smallest overall inertia.

English