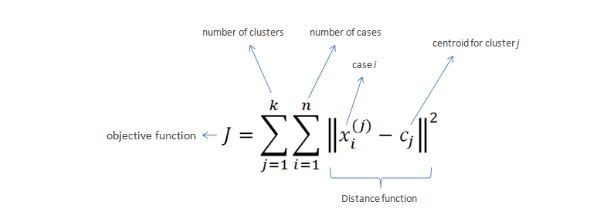
K-means algorithm

When we have an unlabeled data set, the first algorithm that strikes us is the k-means clustering. It is fast and efficient in terms of computational cost.

When we apply k-means on the data then to make n clusters, random points are initialized as cluster centroids. Each data point is assigned to the nearest cluster center. Then the centroid is updated by the average value of all points in each cluster independently. The process repeats as long as the centroid location changes. We are trying to find the minimum distance between all points implying that the data points have been separated such that they form the most compact clusters with the least variance between them.

Therefore, we can say that the objective of K-means clustering is to minimize total intra cluster variance. The variance is a squared error function.



We use the squared error functions because it has nicer mathematical properties than the absolute error function.

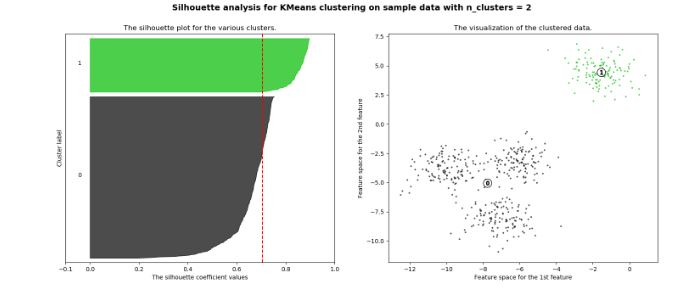
The main focus is to choose the number of clusters n. The maximum accuracy is assigned by each data point being assigned to its own cluster. Or the maximum compression of data is obtained by everything being in a single cluster.

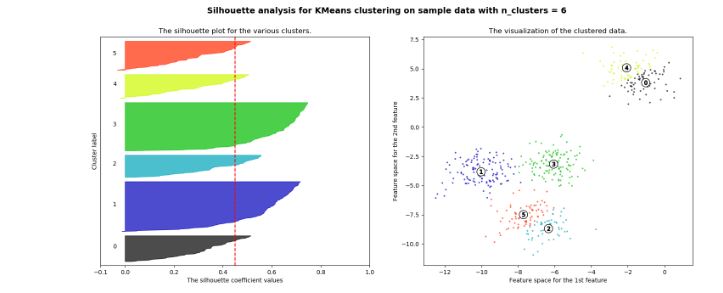
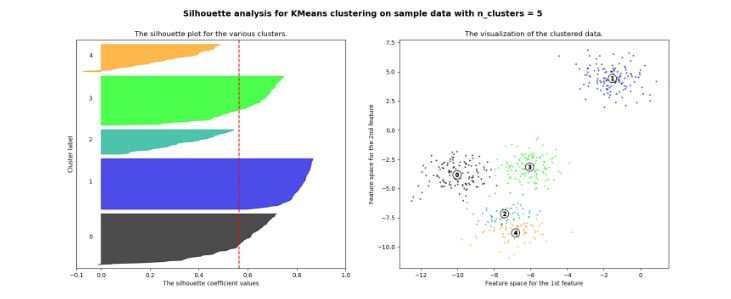
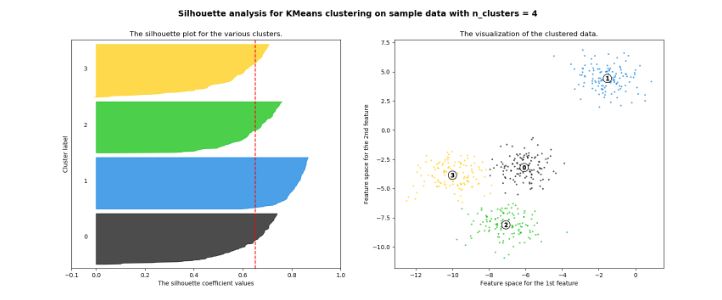
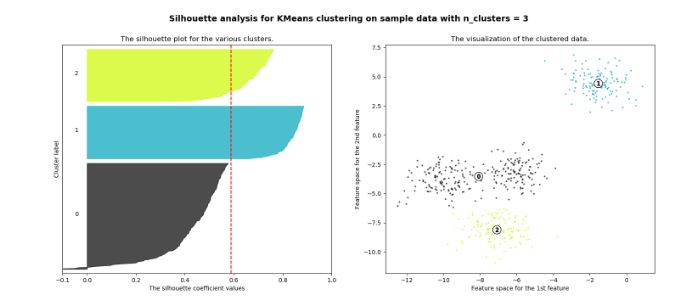
The criterion that gets minimized is called the inertia or within-cluster sum-of-squares.

**Elbow Method:**

To determine the number of clusters we look at a graph of the percentage of variation explained based on the number of clusters. If one plots the percentage of variation explained with respect to the number of clusters, the first few clusters will add much information but at some point, the marginal gain will drop, giving an angle in the graph. The number of clusters is chosen at this point, hence the elbow criterion. Percentage of variation explained is given by the ratio of between the group variation and total variation. This is also known as the f-test. A variation of the method is within the group variance.

**Silhouette analysis:**

It is used to study the separation distance between the resulting clusters. The plot displays a measure of how close each point in one cluster is to points in the neighboring cluster. This measure has a range of [-1,1]. Silhouette coefficients with a value of 1 means that the points are far away from neighboring cluster. If the value is 0 then that means the point is right at the decision boundary between the two clusters. The negative values can indicate that the points are assigned to the wrong clusters. 



These are diagrams of the silhouette plots with different numbers of clusters. The plot shows how the cluster area changes according to the number of clusters chosen. For e.g., with 2 clusters, the 0 cluster is extremely big and then with 4 clusters it gets divided into 4 equal areas of cluster. The average cluster score is bad for cluster number 3, 5 and 6 definitely. It is lower than even the average score.

For n\_clusters = 2 The average [silhouette\_score](http://scikit-learn.org/stable/modules/generated/sklearn.metrics.silhouette_score.html#sklearn.metrics.silhouette_score) **is** : 0.704978749608

For n\_clusters = 3 The average [silhouette\_score](http://scikit-learn.org/stable/modules/generated/sklearn.metrics.silhouette_score.html#sklearn.metrics.silhouette_score) **is** : 0.588200401213

For n\_clusters = 4 The average [silhouette\_score](http://scikit-learn.org/stable/modules/generated/sklearn.metrics.silhouette_score.html#sklearn.metrics.silhouette_score) **is** : 0.650518663273

For n\_clusters = 5 The average [silhouette\_score](http://scikit-learn.org/stable/modules/generated/sklearn.metrics.silhouette_score.html#sklearn.metrics.silhouette_score) **is** : 0.563764690262

For n\_clusters = 6 The average [silhouette\_score](http://scikit-learn.org/stable/modules/generated/sklearn.metrics.silhouette_score.html#sklearn.metrics.silhouette_score) **is** : 0.450466629437

**Limitations of k-means algorithm:**

1. The clusters are of unequal size or density.
2. The clusters are non-spherical. The inertia makes an assumption that the clusters are convex and isotropic. It responds poorly with elongated clusters or manifolds with irregular shapes.
3. There are outliers in the data.
4. Inertia is not a normalized metric. So, in higher dimensions, Euclidean distance does not hold up. They tend to become inflated and therefore, in these cases, running dimensionality reduction techniques like PCA is very useful.

**Evaluation of k-means result:**

If we have 2-D data, we can look at the plot and visually check the quality of the result that our algorithm gives however, it is difficult to do the same when the data is in higher dimensions. Otherwise, we have two different types of evaluation: supervised and unsupervised.

**Supervised evaluation:**

If we have a pre-classified data set that can act as a benchmark, then we can check the results of the k-means algorithm. Typically, how it works is that an expert would carefully process a small dataset assigning data points to clusters manually. This is the gold standard for evaluation purposes and tells us how close the results are to the benchmark.

Using conditional entropy analysis, we can come up with metrics to measure the performance. Two desirable objectives for cluster assignment are:

Homogeneity: each cluster has members from only a single class

Completeness: All members of a given class are assigned to the same cluster

Homogeneity and completeness scores are bounded by 0 and 1.

**>>> from** **sklearn** **import** metrics

**>>>** labels\_true = [0, 0, 0, 1, 1, 1]

**>>>** labels\_pred = [0, 0, 1, 1, 2, 2]

**>>>** metrics.homogeneity\_score(labels\_true, labels\_pred)

0.66...

**>>>** metrics.completeness\_score(labels\_true, labels\_pred)

0.42...

Their harmonic mean called **V-measure** is computed by [**v\_measure\_score**](http://scikit-learn.org/stable/modules/generated/sklearn.metrics.v_measure_score.html#sklearn.metrics.v_measure_score):

**>>>** metrics.v\_measure\_score(labels\_true, labels\_pred)

0.51...

The V-measure is actually equivalent to the mutual information (NMI) discussed above normalized by the sum of the label entropies.

Homogeneity, completeness and V-measure can be computed at once using **[homogeneity\_completeness\_v\_measure](http://scikit-learn.org/stable/modules/generated/sklearn.metrics.homogeneity_completeness_v_measure.html" \l "sklearn.metrics.homogeneity_completeness_v_measure" \o "sklearn.metrics.homogeneity_completeness_v_measure)** as follows:

**>>>** metrics.homogeneity\_completeness\_v\_measure(labels\_true, labels\_pred)

(0.66..., 0.42..., 0.51...)

The following clustering assignment is slightly better, since it is homogeneous but not complete:

**>>>** labels\_pred = [0, 0, 0, 1, 2, 2]

**>>>** metrics.homogeneity\_completeness\_v\_measure(labels\_true, labels\_pred)

(1.0, 0.68..., 0.81...)

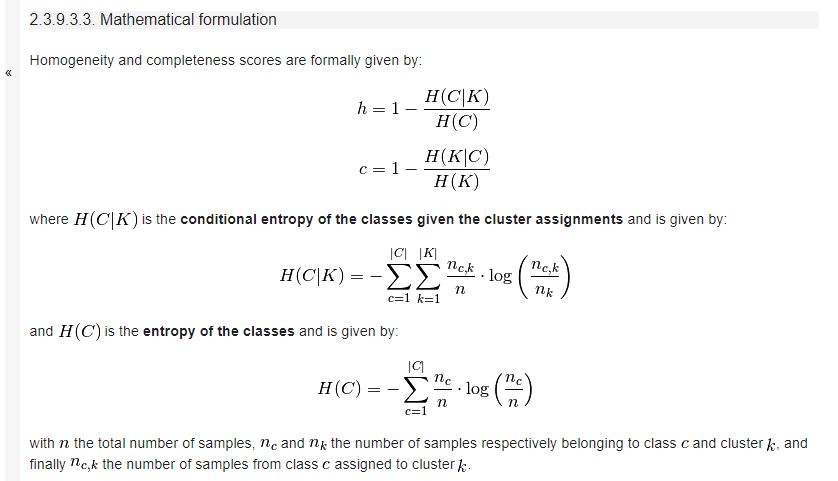
**Note:**

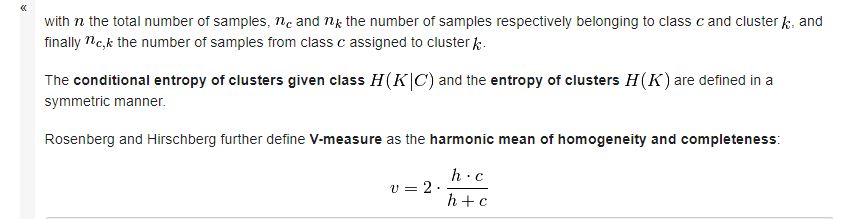
**[v\_measure\_score](http://scikit-learn.org/stable/modules/generated/sklearn.metrics.v_measure_score.html" \l "sklearn.metrics.v_measure_score" \o "sklearn.metrics.v_measure_score)** is **symmetric**: it can be used to evaluate the **agreement** of two independent assignments on the same dataset.

This is not the case for **[completeness\_score](http://scikit-learn.org/stable/modules/generated/sklearn.metrics.completeness_score.html" \l "sklearn.metrics.completeness_score" \o "sklearn.metrics.completeness_score)** and **[homogeneity\_score](http://scikit-learn.org/stable/modules/generated/sklearn.metrics.homogeneity_score.html" \l "sklearn.metrics.homogeneity_score" \o "sklearn.metrics.homogeneity_score)**: both are bound by the relationship: homogeneity\_score(a, b) == completeness\_score(b, a)

The advantages to this method is that there are bounded scores. So, we know that 0 is the least and 1 is as perfect as it can get. Intuitively, from the value of V-measure we can know the kind of mistakes being made by the algorithm. There is no assumption being made on the cluster structure. It works with k-means and with spectral clustering as well.

The drawbacks on the other hand are as follows: the metrics are not normalized with respect to random labeling. Hence, depending on the number of samples, clusters and ground truth classes, a complete random labeling will not yield the same values for homogeneity, completeness and v-measure. This is essentially because, random labeling does not give a score of 0 for a large number of samples. This holds true as long as the number of clusters is less than 10 and the samples greater than 1000. In the case that the number of samples is less and the clusters more, an adjusted index, like the Adjusted Rand Index (ARI) is used.





**Unsupervised Evaluation:**

We can just use the inter cluster and intra cluster variance to decide if the results were good enough. They essentially describe the cohesion of the points that are within a cluster and the separation between the clusters. We want well cohesive clusters that are properly separated from each other.

Even though k-means offers so many advantages, there is something called the Random Initialization Trap that fails the algorithm. The selection of the centroids in the very beginning can determine the success of the algorithm.