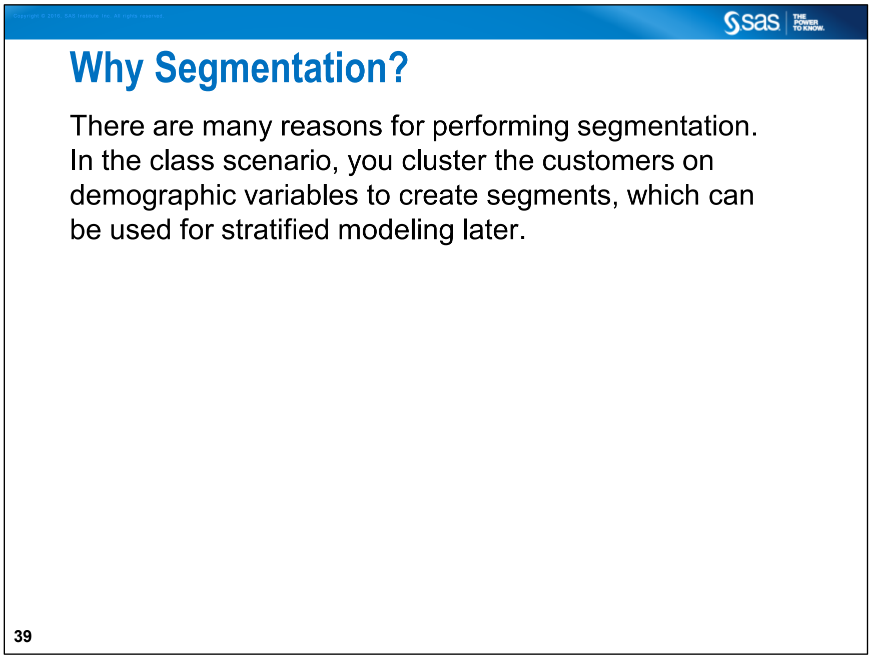
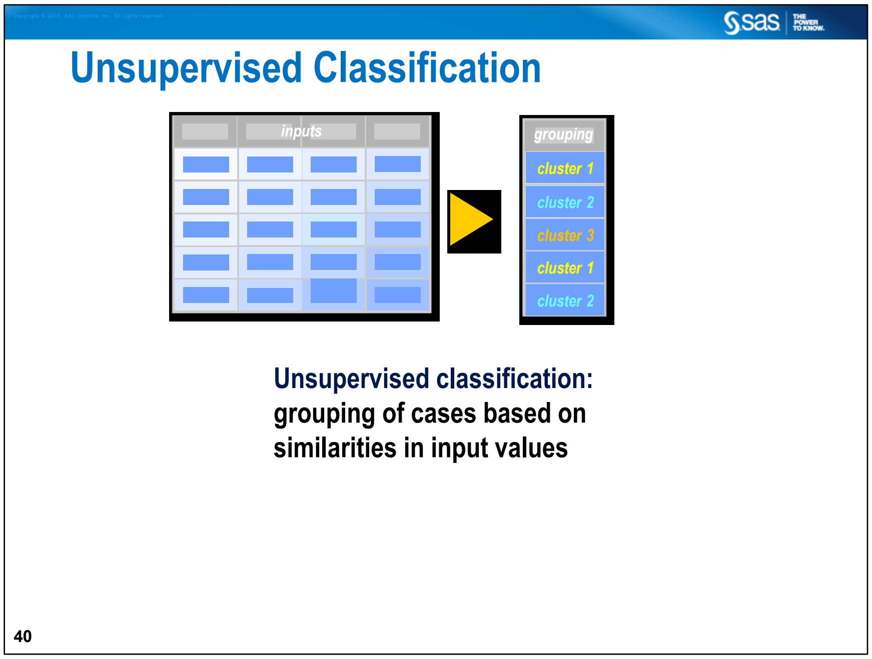
Segmentation Ideas and SAS Visual Statistics Cluster Details (Self-Study)

Segmentation Concepts



Why Segmentation?

There are many reasons for performing segmentation. In the class scenario, you cluster the customers on demographic variables to create segments, which can be used for stratified modeling later.



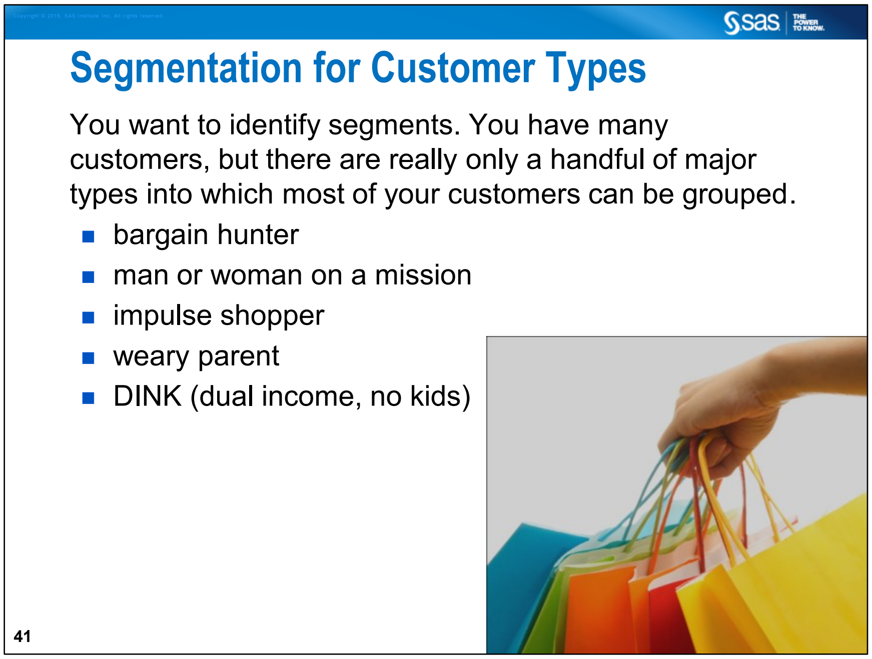
Unsupervised Classification

Unsupervised classification: grouping of cases based on similarities in input values

Cluster analysis is a form of unsupervised classification that attempts to group cases in the data based on similarities in ***input*** variables. It is a data-reduction method because an entire training data set can be represented by a small number of clusters. The groupings are known as *clusters* or *segments*, and they can be applied to other data sets to classify new cases. Unsupervised classification is distinguished from *supervised classification* (in which there is a known criterion).

The purpose of clustering is often description. For example, segmenting existing customers into groups and associating a distinct profile with each group might help future marketing strategies. However, there is no guarantee that the resulting clusters are meaningful or useful.

Clustering can be useful as a preliminary step in predictive modeling. For example, customers can be clustered into homogeneous groups based on sales of different items. Then, the clusters can be used as stratification variables for building predictive models separately on each group.

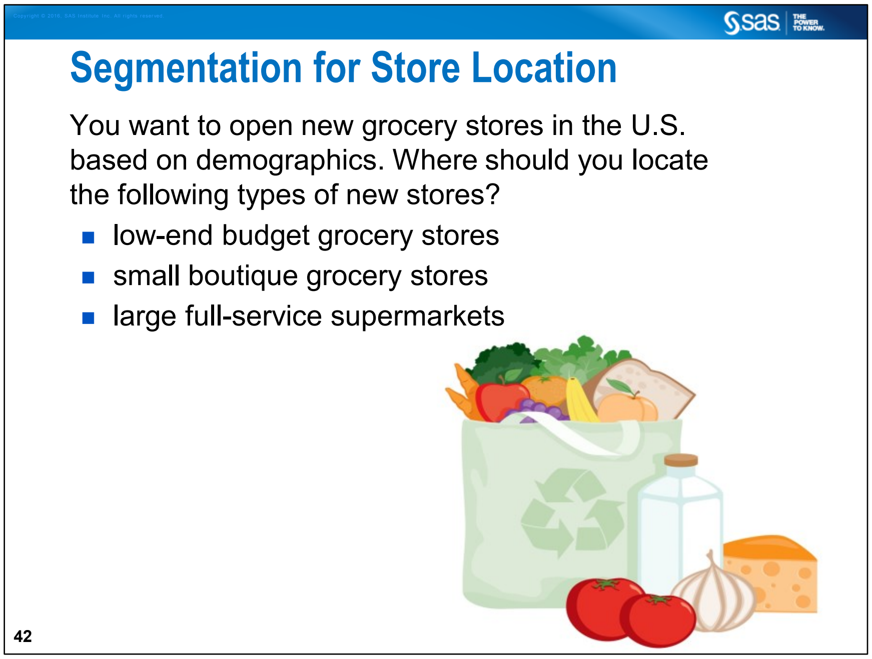


Segmentation for Customer Types

You want to identify segments. You have many customers, but there are really only a handful of major types into which most of your customers can be grouped.

- bargain hunter
- man or woman on a mission
- impulse shopper
- weary parent
- DINK (dual income, no kids)

41

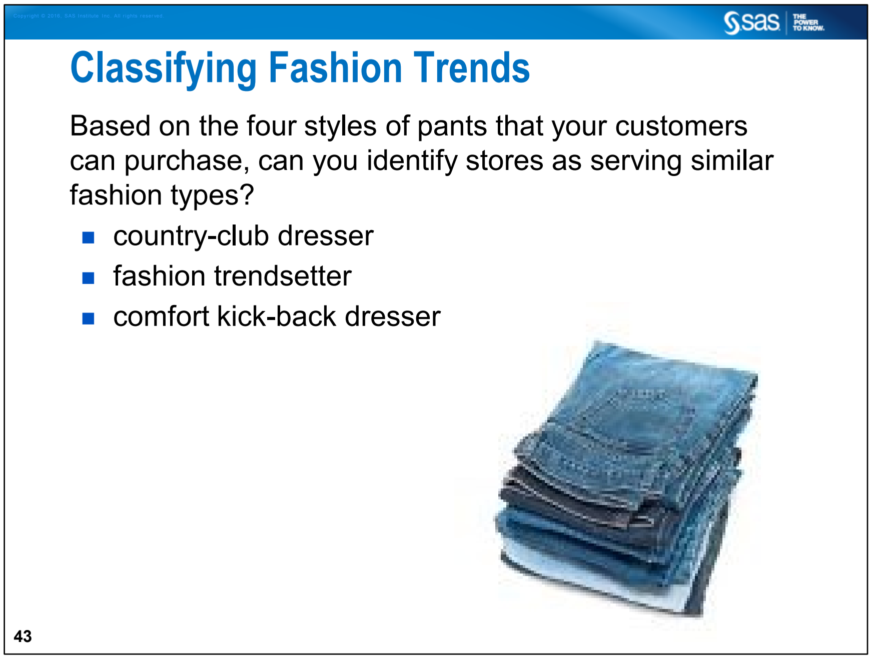


Segmentation for Store Location

You want to open new grocery stores in the U.S. based on demographics. Where should you locate the following types of new stores?

- low-end budget grocery stores
- small boutique grocery stores
- large full-service supermarkets

42



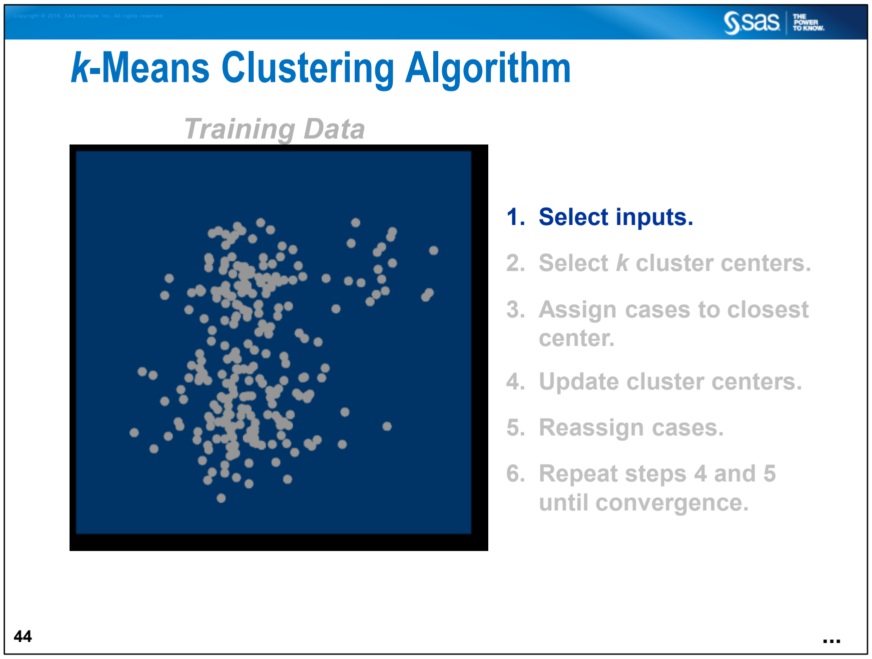
Classifying Fashion Trends

Based on the four styles of pants that your customers can purchase, can you identify stores as serving similar fashion types?

- country-club dresser
- fashion trendsetter
- comfort kick-back dresser

43

SAS Visual Statistics: Clustering Method Details



k-Means Clustering Algorithm

Training Data

1. Select inputs.
2. Select *k* cluster centers.
3. Assign cases to closest center.
4. Update cluster centers.
5. Reassign cases.
6. Repeat steps 4 and 5 until convergence.

44

...

One of the most commonly used methods for clustering is the *k-means algorithm*. It is a straightforward algorithm that scales well to large data sets.

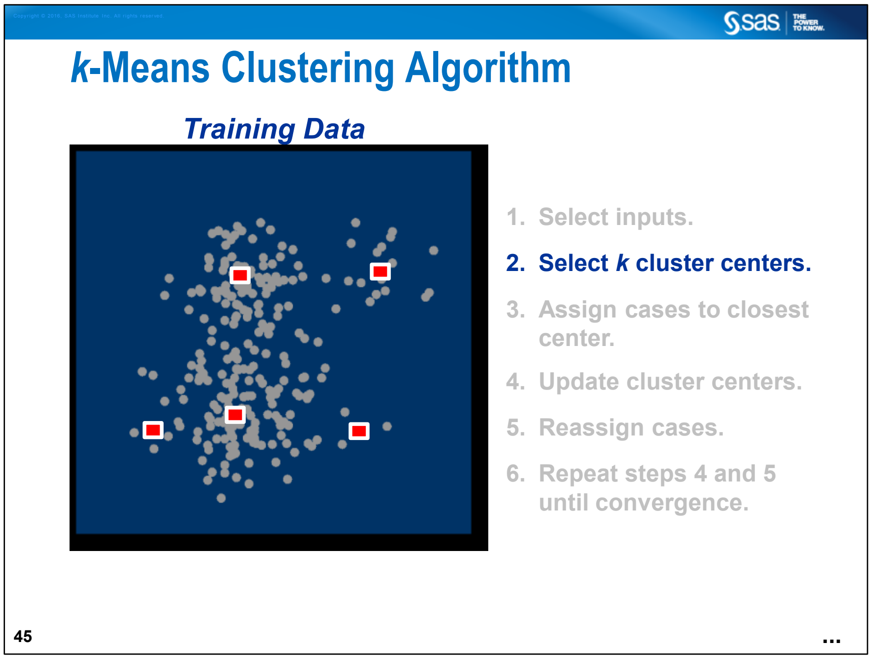
Although it is often overlooked as an important part of a clustering process, the first step in using the *k*-means algorithm is to choose a set of inputs. In general, you should seek inputs that have these attributes:

- are meaningful to the analysis objective
- are relatively independent
- are limited in number
- have a measurement level of *Interval*
- have low kurtosis and skewness (at least in the training data)

✎ *Skewness* is a measure of the symmetry of a distribution. Skewness far from 0 indicates asymmetric data. *Kurtosis* is a measure of the density of a distribution in the peak, flanks, and tails. A symmetric distribution with kurtosis near 0 (or 3 in some software) has the characteristic bell-shaped curve of the normal distribution. Kurtosis can easily be interpreted if there is no symmetry. See DeCarlo (1997)[1] for more information about the impact and meaning of kurtosis in data analysis.

Choosing meaningful inputs is clearly important for the interpretation and explanation of the generated clusters. Independence and limited input count make the resulting clusters more stable. An interval measurement level is recommended for *k*-means to produce nontrivial clusters. Low kurtosis and skewness statistics on the inputs avoid creating single-case outlier clusters.

[1] DeCarlo, L. T. 1997. “On the meaning and use of kurtosis.” *Psychological Methods* 2:292-307.



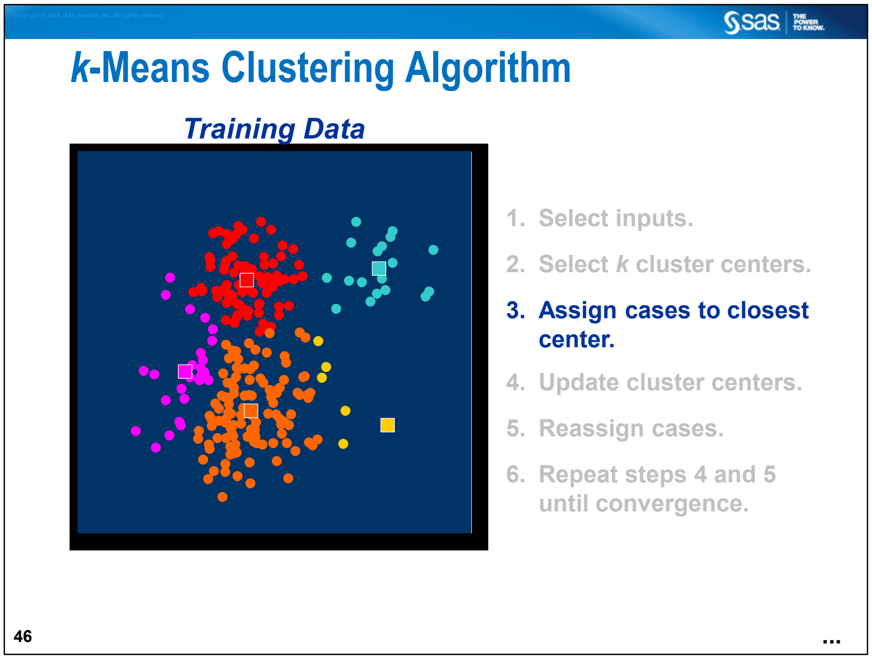
k-Means Clustering Algorithm

Training Data

1. Select inputs.
2. **Select *k* cluster centers.**
3. Assign cases to closest center.
4. Update cluster centers.
5. Reassign cases.
6. Repeat steps 4 and 5 until convergence.

45 ...

The next step in the *k*-means algorithm is to choose a value for *k*, the number of cluster centers. SAS Visual Statistics uses five clusters by default, although you can change this easily. You should choose *k* to be consistent with the natural concentrations of cases, or with your analytic objectives. For example, if you are interested in promoting three offerings, then three (or more) clusters would be most useful.



k-Means Clustering Algorithm

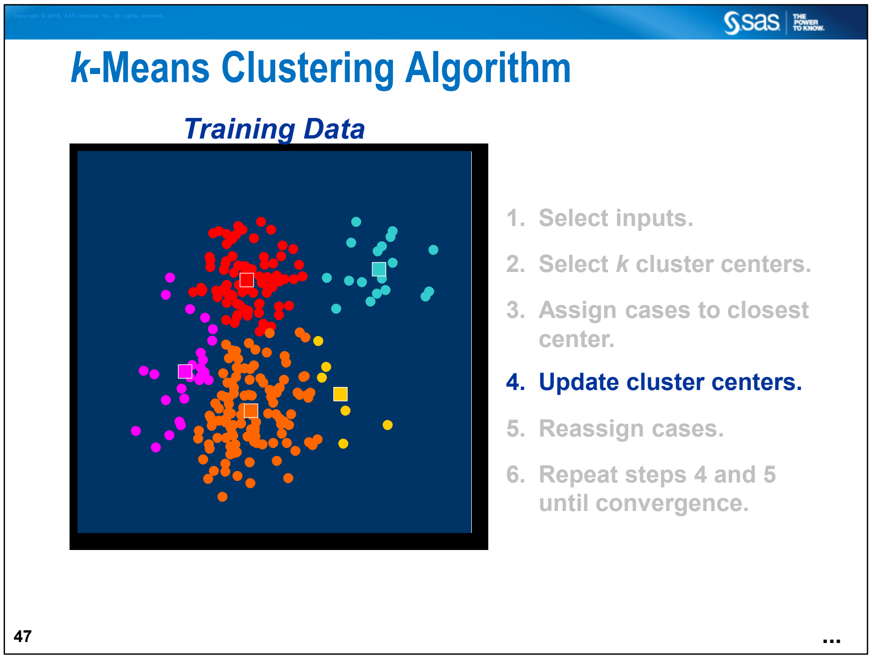
Training Data

1. Select inputs.
2. Select *k* cluster centers.
3. **Assign cases to closest center.**
4. Update cluster centers.
5. Reassign cases.
6. Repeat steps 4 and 5 until convergence.

46 ...

The Euclidean distance from each case in the training data to each cluster center is calculated. Cases are assigned to the closest cluster center.

The *Euclidean distance* is the distance between two points measured as a straight line. It is computed based on the Pythagorean Theorem and does not directly account for differences in scale of measurement. Because the distance metric is Euclidean, it is important for the inputs to have compatible measurement scales. Unexpected results can occur if one input's measurement scale differs greatly from the others.



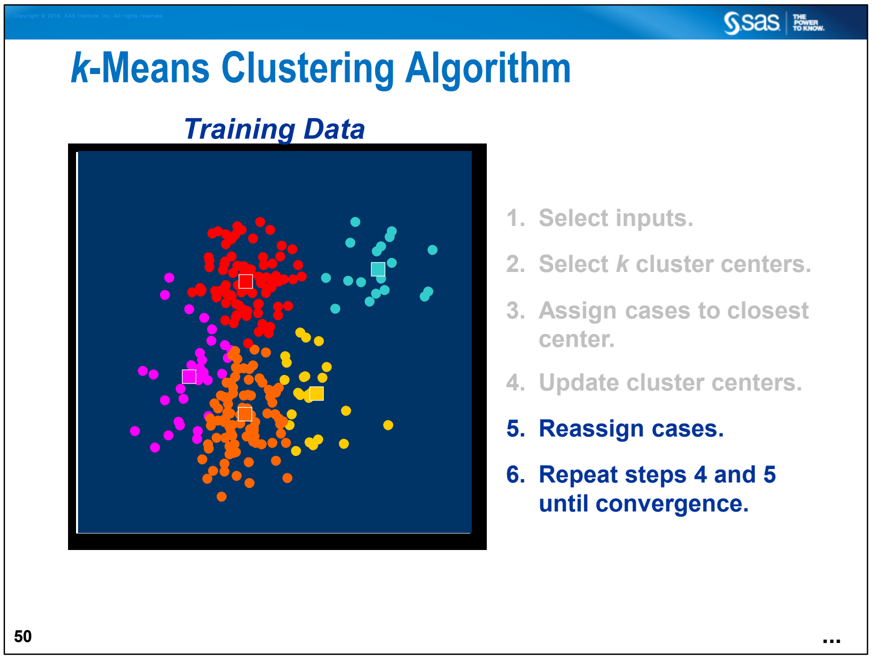
k-Means Clustering Algorithm

Training Data

1. Select inputs.
2. Select *k* cluster centers.
3. Assign cases to closest center.
4. **Update cluster centers.**
5. Reassign cases.
6. Repeat steps 4 and 5 until convergence.

47 ...

The cluster centers are updated to equal the average of the cases assigned to the cluster in the previous step.



k-Means Clustering Algorithm

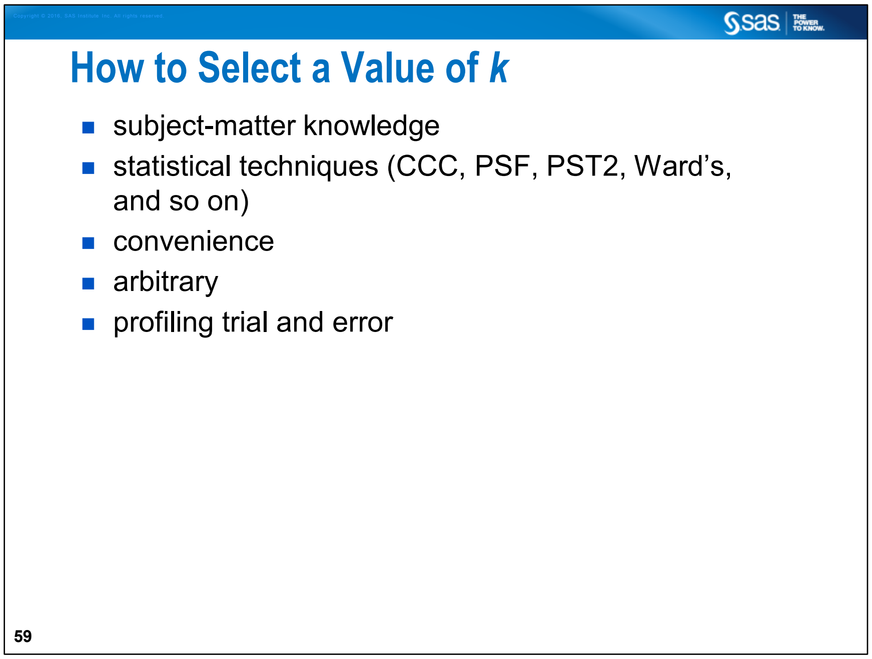
Training Data

1. Select inputs.
2. Select *k* cluster centers.
3. Assign cases to closest center.
4. Update cluster centers.
5. **Reassign cases.**
6. **Repeat steps 4 and 5 until convergence.**

50 ...

Cases are reassigned to the closest cluster center.

The update and reassign steps are repeated until the process converges. On convergence, final cluster assignments are made. Each case is assigned to a unique segment. The segment definitions can be stored and applied to new cases outside of the data used to develop clusters.

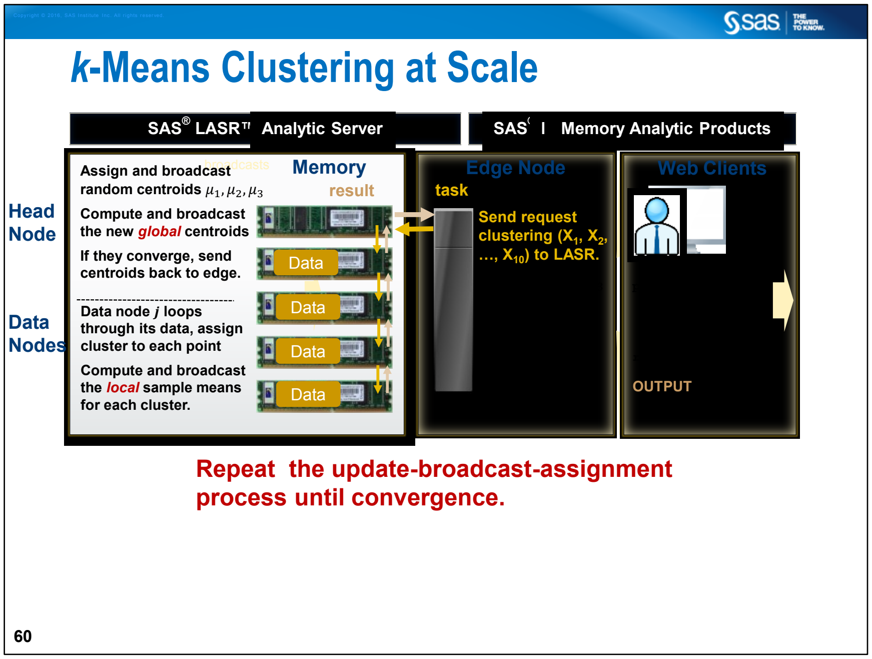


How to Select a Value of *k*

- subject-matter knowledge
- statistical techniques (CCC, PSF, PST2, Ward's, and so on)
- convenience
- arbitrary
- profiling trial and error

59

There are many techniques, from the scientifically rigorous to the arbitrary, for determining the number of clusters, or *k*. In SAS Visual Statistics, you can specify the number of clusters, profile that selection, and try other candidate solutions. The default initial number of clusters is 5.



k-Means Clustering at Scale

Assign and broadcast random centroids μ_1, μ_2, μ_3

Compute and broadcast the new *global* centroids

If they converge, send centroids back to edge.

Data node *j* loops through its data, assign cluster to each point

Compute and broadcast the *local* sample means for each cluster.

Send request clustering (X_1, X_2, …, X_{10}) to LASR.

Repeat the update-broadcast-assignment process until convergence.

60

If you use SAS Visual Statistics to perform clustering in a distributed environment, then the *k*-means algorithm is modified somewhat to take advantage of multiple compute and data nodes. For example, suppose you are running in a distributed environment with data in Hadoop.

1. Submit code that asks for *k*-means clustering of 10 variables on a data set named **dat1**. The number of clusters is 3.
2. The Edge node sends the request to the server and waits for the results.
3. The Head node randomly initializes three centroids and broadcasts them to the data nodes. In this case, the data structure being broadcast is actually a 3-by-10 array.
4. Each of the data nodes goes through its own local data, and assigns each observation to the closest centroid.
5. The local cluster sample means are then calculated on each data node, and this information is sent back to the Head node. Again, the data structure being broadcast is only the 3-by-10 array.
6. The cluster sample means for the entire data, referred to as ***global centroids*** here, are aggregated to the Head node.
7. Then, the new clusters are broadcast back to the data nodes for the next round of cluster assignment.

Repeat this update-broadcast-assignment process until convergence. After the algorithm converges, the results are sent back to the Edge node and then to the web client for final output.

What is described here is one of many possible *reduce* strategies that can be implemented with the SAS LASR Analytic Server. There are others that are more efficient, although they are perhaps less convenient, for teaching purposes, than returning the results to the Head node repeatedly. The reduce method used by the LASR Analytic Server depends on the specific problem that the LASR Analytic Server is solving.