Chapter 11

Unsupervised Learning
Clustering

Unsupervised Learning

Types of Machine Learning

Supervised	data points have known outcome
Unsupervised	data points have unknown outcome

Types of Machine Learning

Supervised	data points have known outcome
Unsupervised	data points have unknown outcome

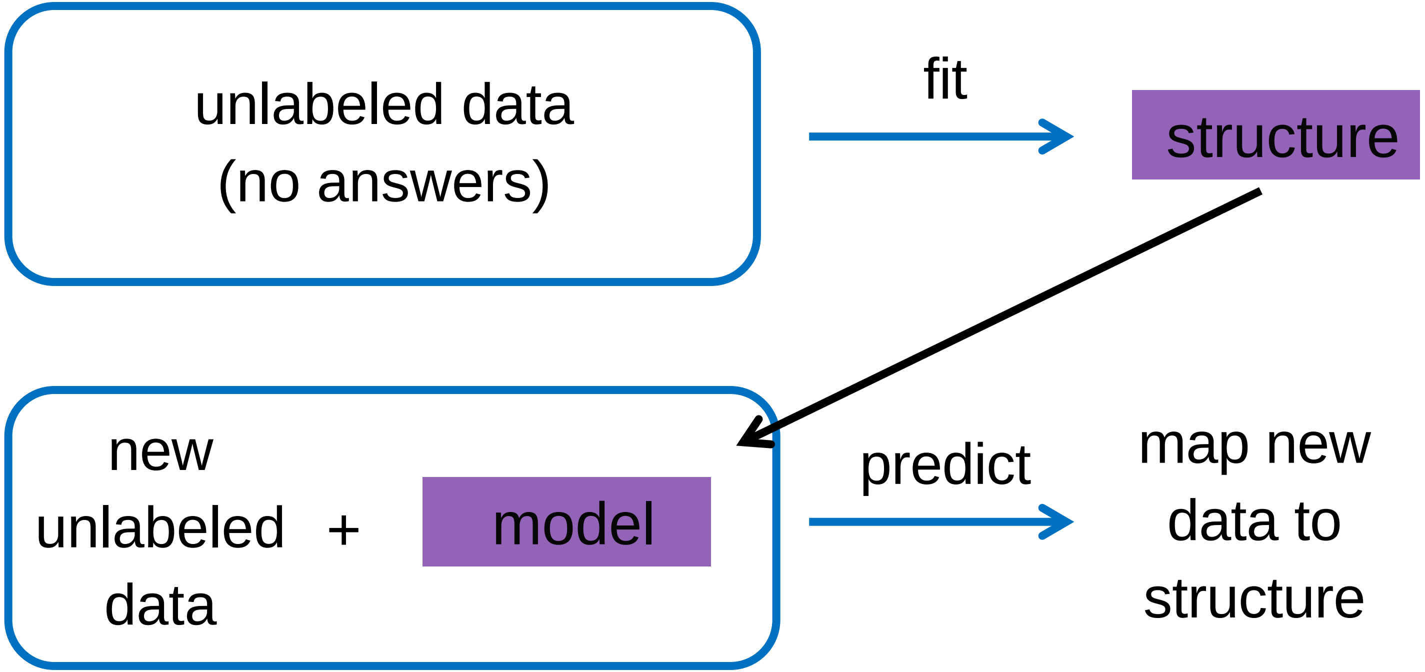
Types of Unsupervised Learning

Clustering — identify unknown structure in data

Types of Unsupervised Learning

Clustering	identify unknown structure in data
Dimensionality Reduction	use structural characteristics to simplify data

Unsupervised Learning Overview



Clustering: Finding Distinct Groups

text articles of unknown topics + model $\xrightarrow{\text{fit}}$ model

text articles of unknown topics + model $\xrightarrow{\text{predict}}$ predict similar articles

Dimensionality Reduction: Simplifying Structure

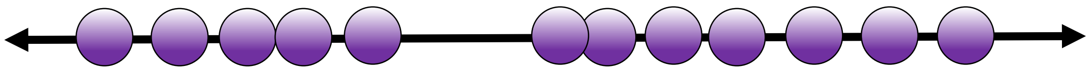
high resolution images + model $\xrightarrow{\text{fit}}$ model

high resolution images + model $\xrightarrow{\text{predict}}$ compressed images

Introduction to Unsupervised Learning

Users of a web application:

One feature (age)



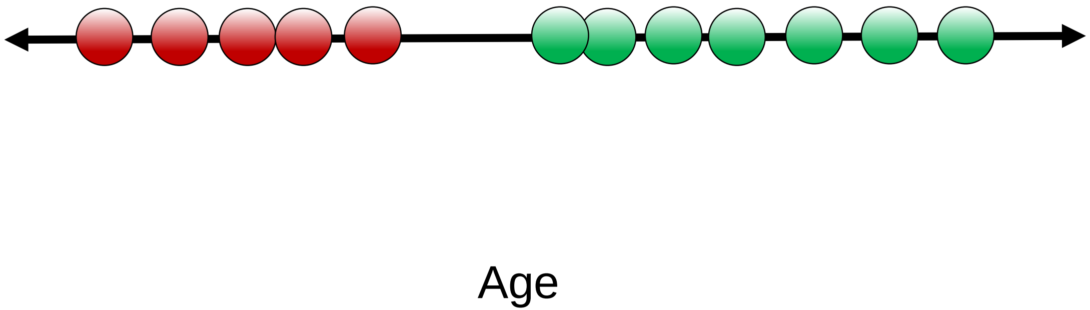
Age

Introduction to Unsupervised Learning

Users of a web application:

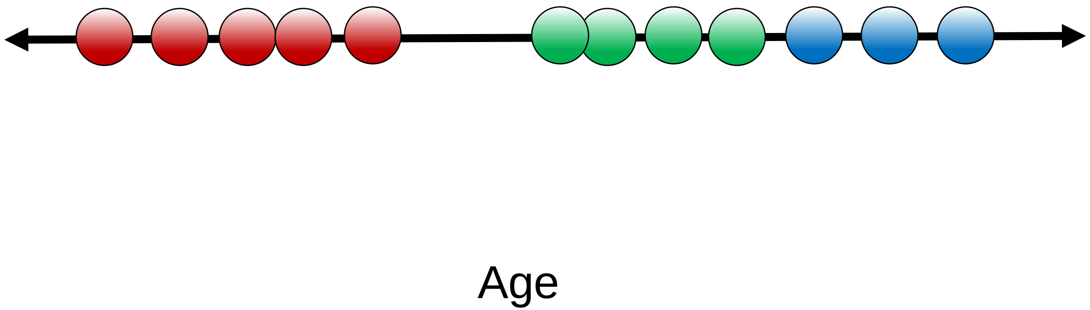
One feature (age)

Two clusters



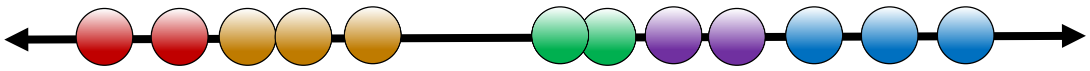
Introduction to Unsupervised Learning

Users of a web application:
One feature (age)
Three clusters



Introduction to Unsupervised Learning

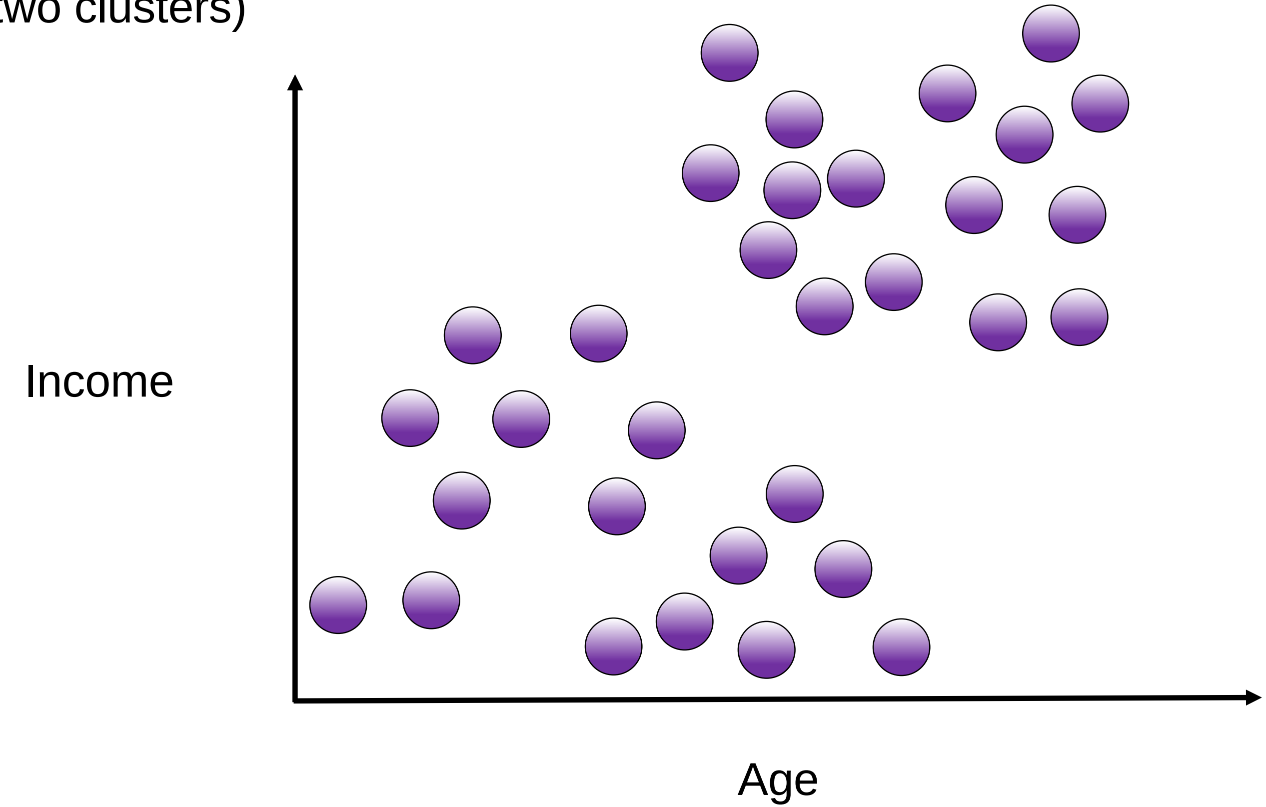
Users of a web application:
One feature (age)
Five clusters



Age

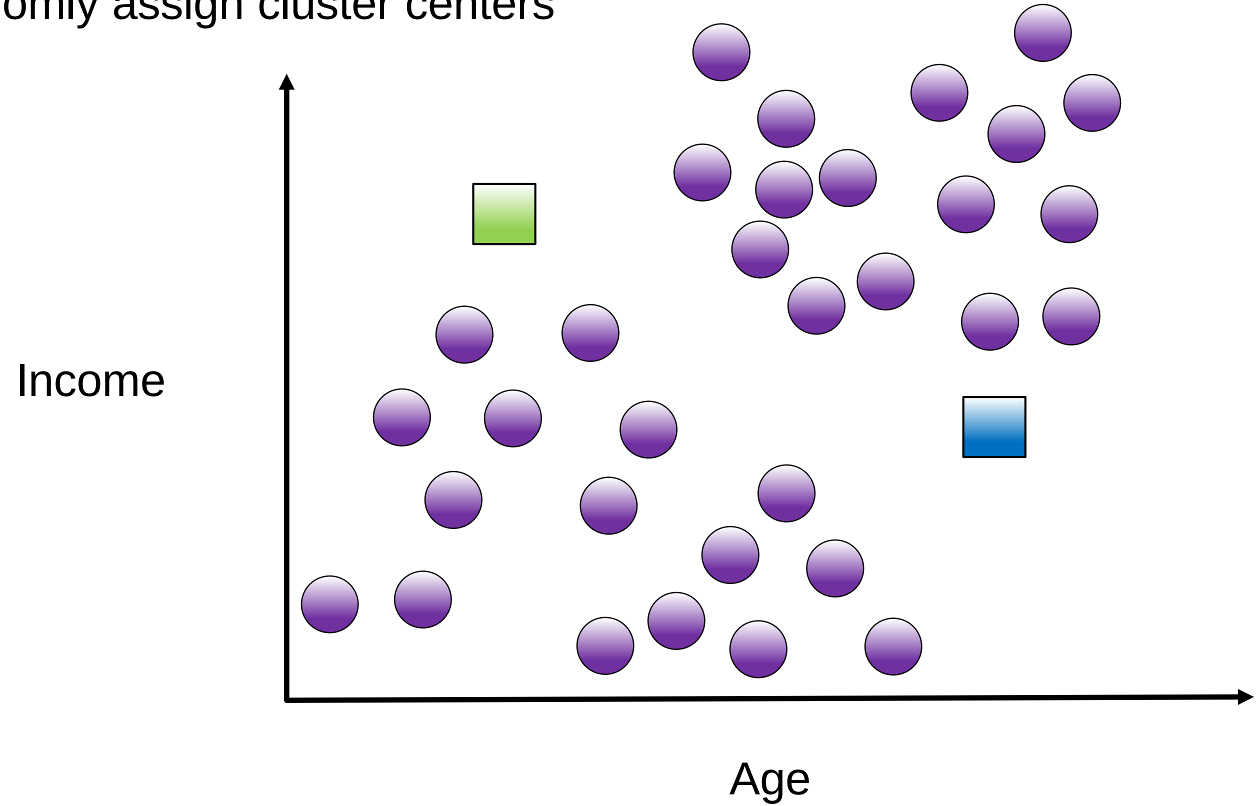
K-Means Algorithm

K = 2 (find two clusters)



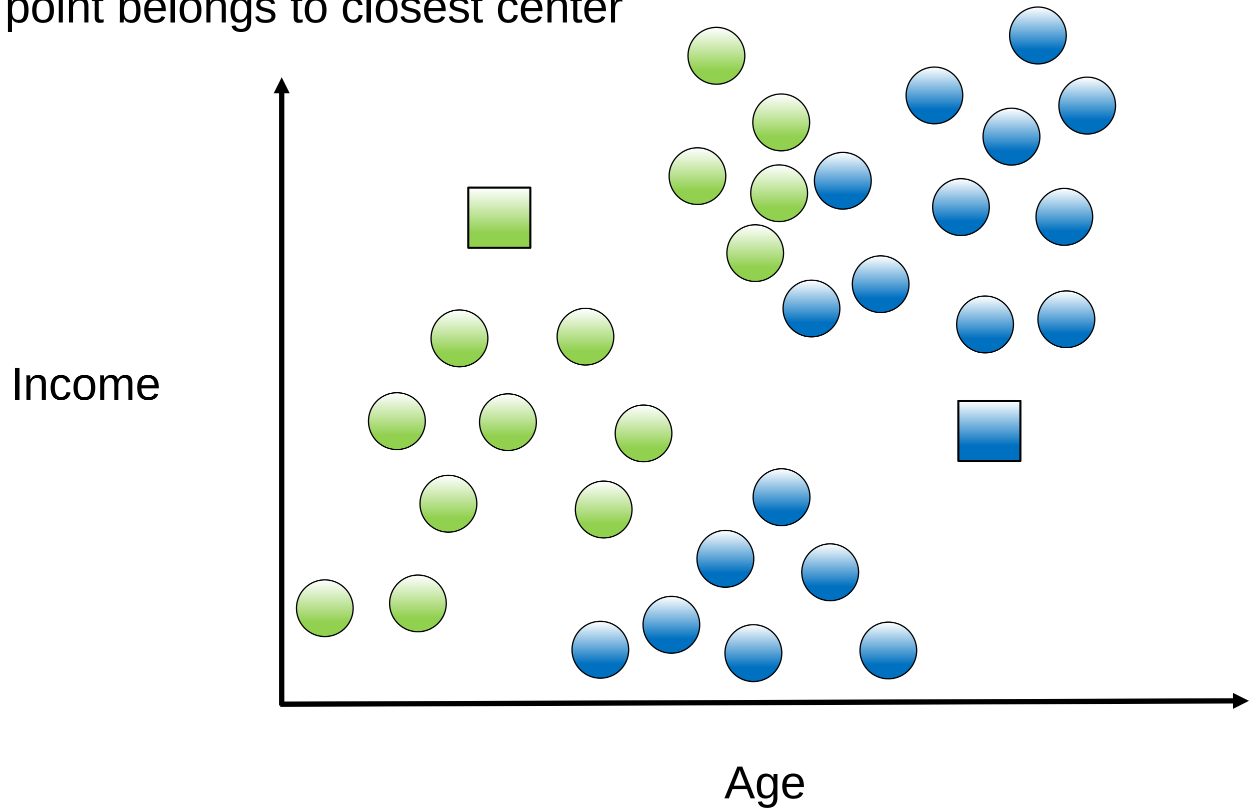
K-Means Algorithm

K = 2, Randomly assign cluster centers



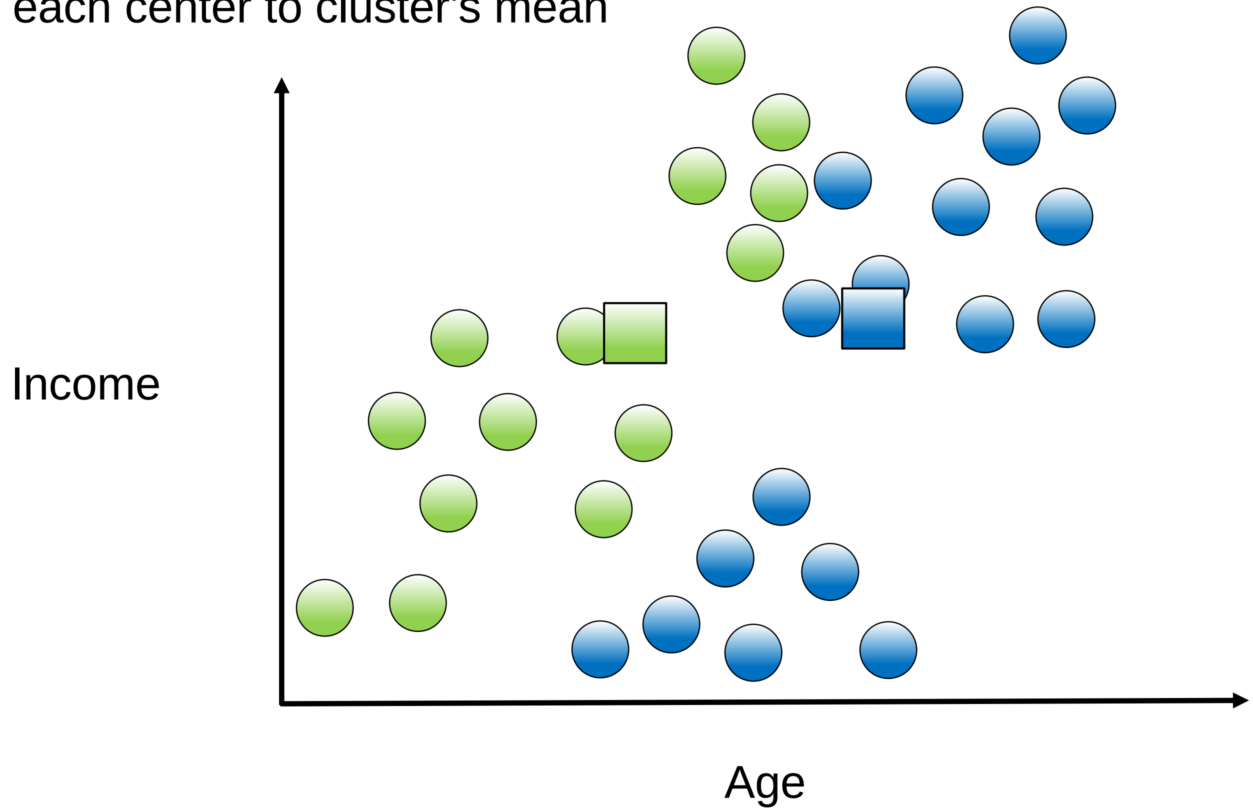
K-Means Algorithm

K = 2, Each point belongs to closest center



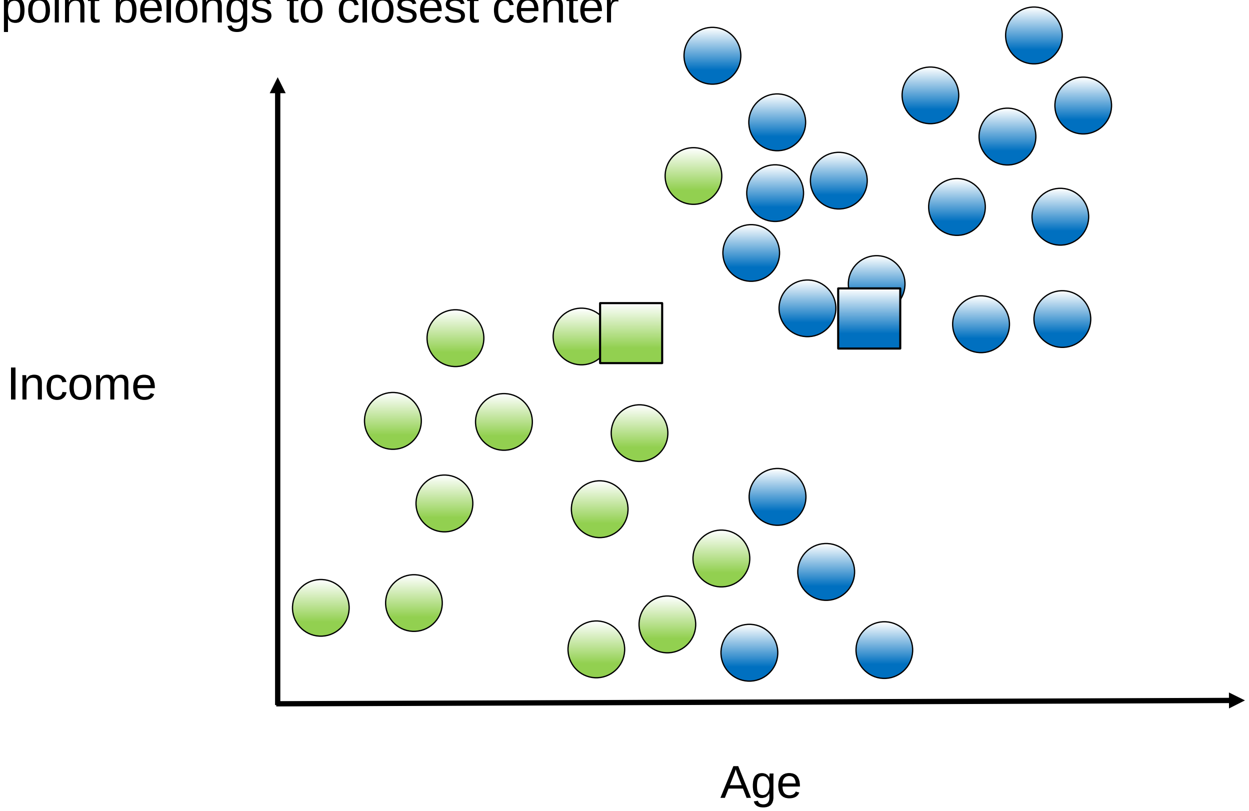
K-Means Algorithm

K = 2, Move each center to cluster's mean



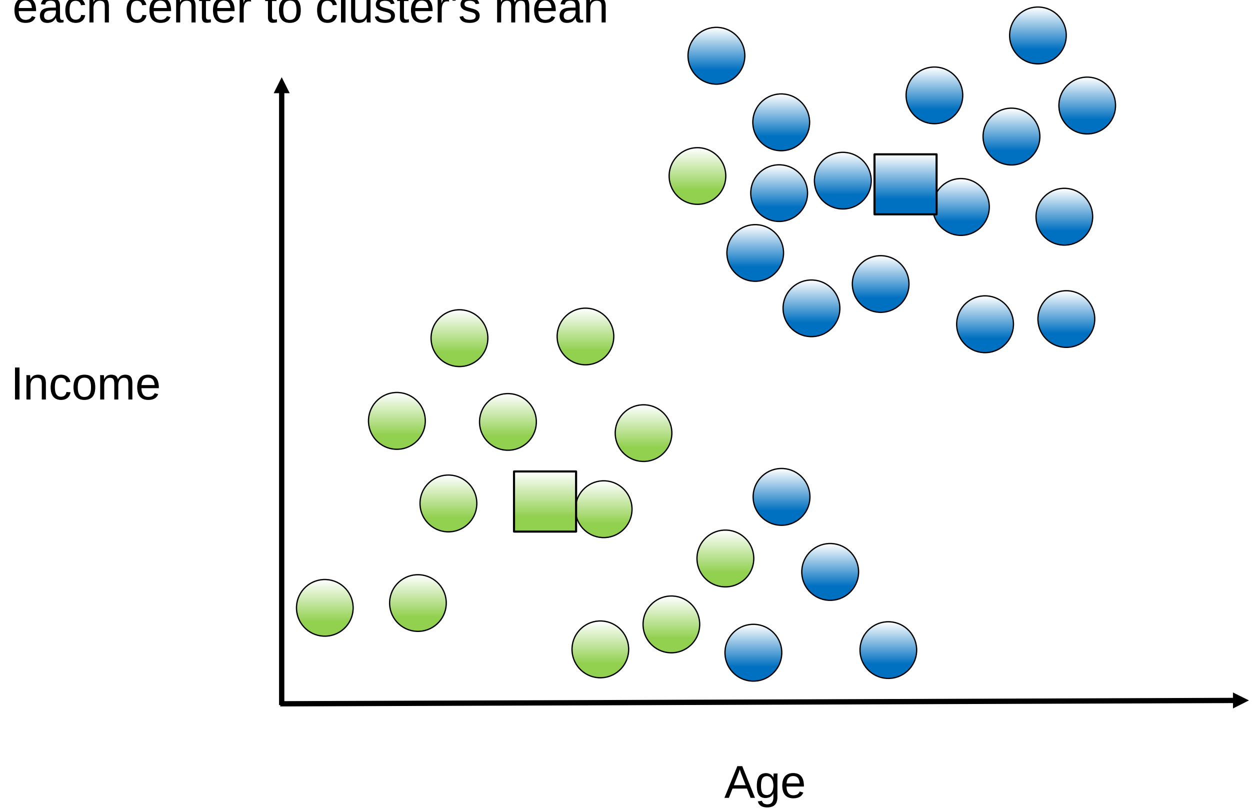
K-Means Algorithm

K = 2, Each point belongs to closest center



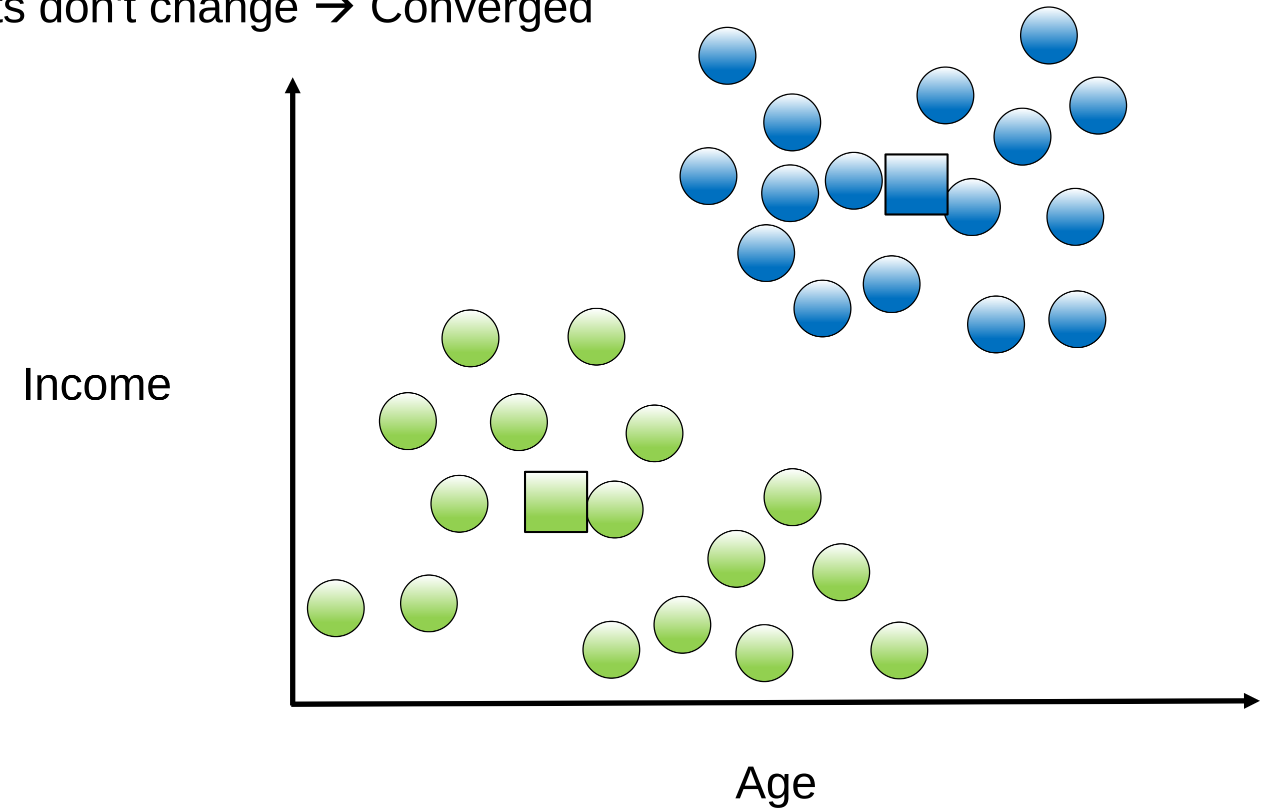
K-Means Algorithm

K = 2, Move each center to cluster's mean



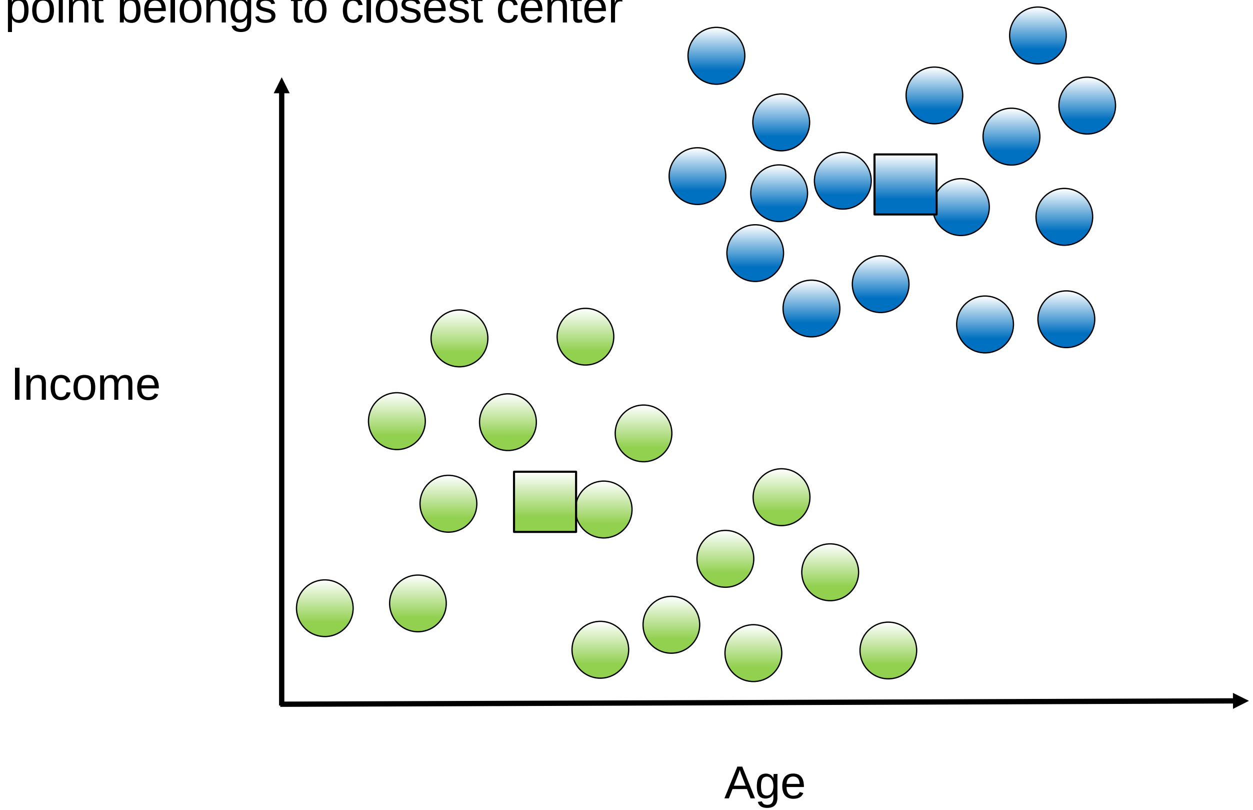
K-Means Algorithm

K = 2, Points don't change → Converged



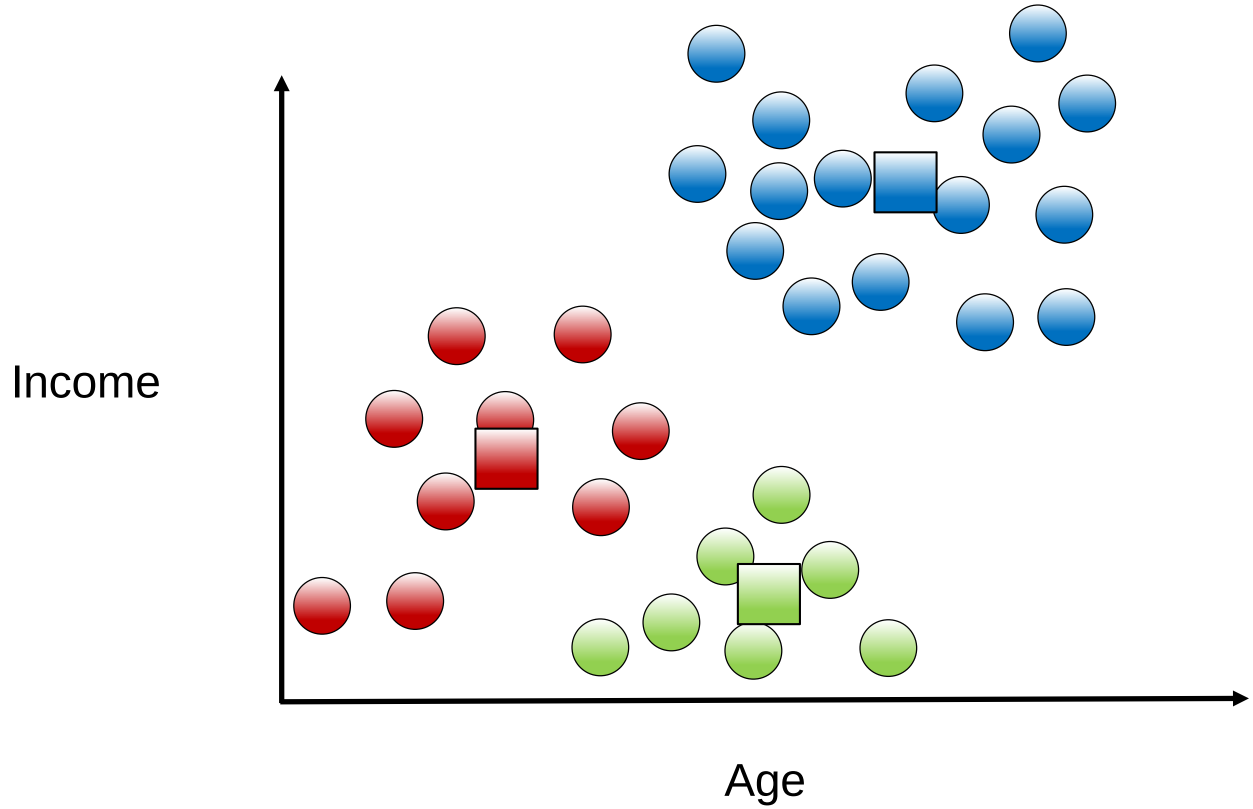
K-Means Algorithm

K = 2, Each point belongs to closest center



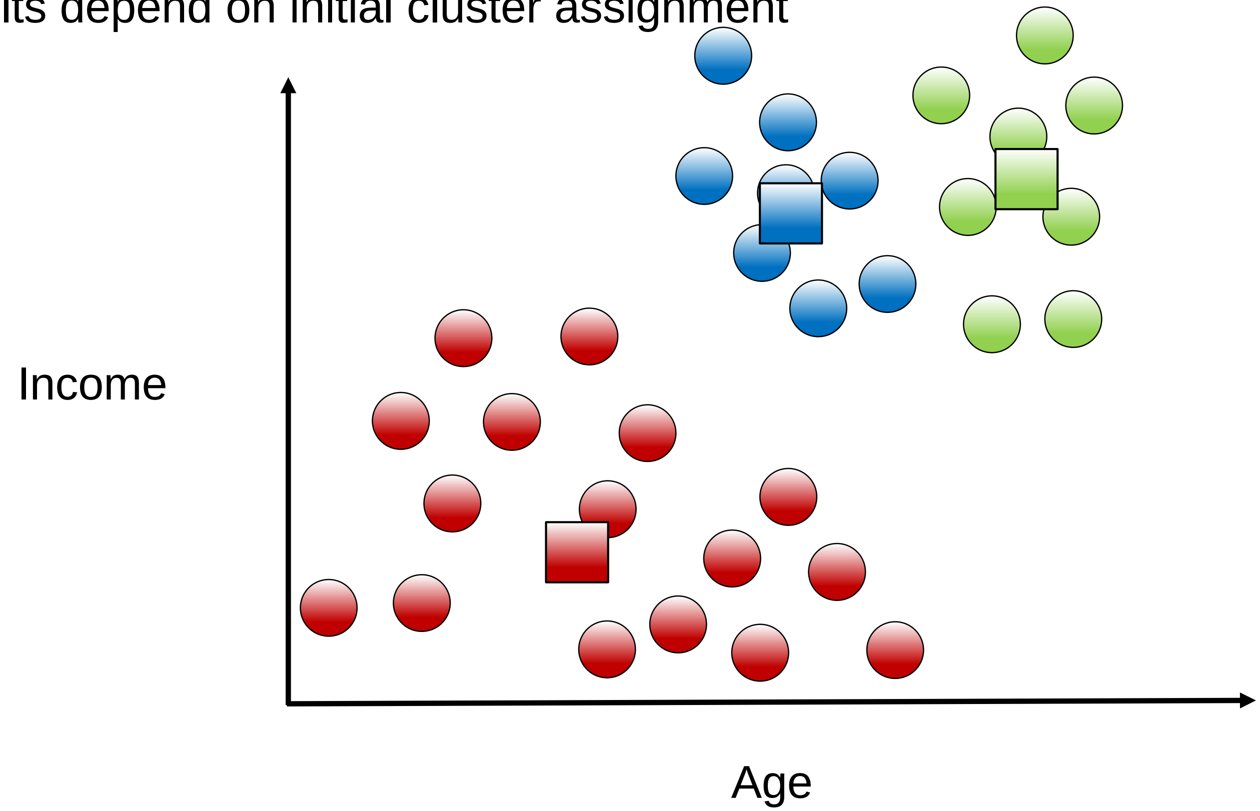
K-Means Algorithm

K = 3

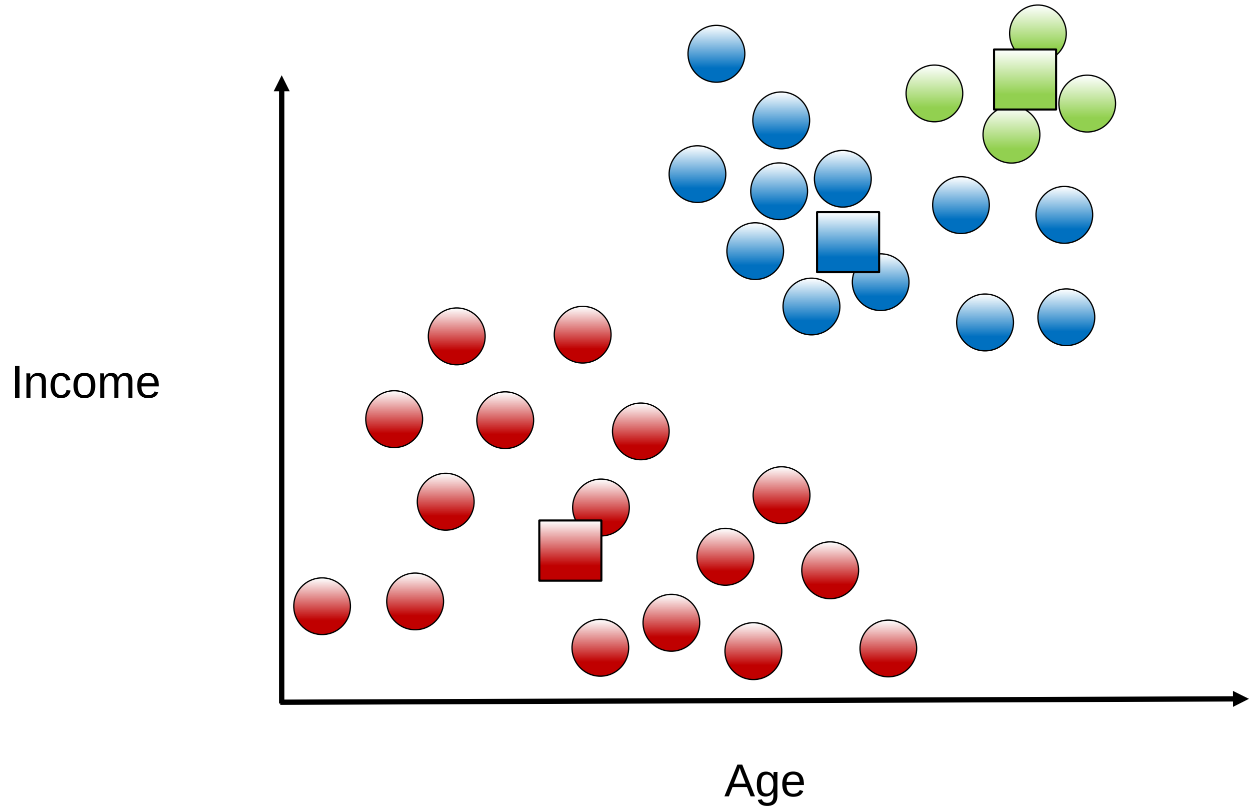


K-Means Algorithm

K = 3, Results depend on initial cluster assignment



Which Model is the Right One?



Which Model is the Right One?

- **Inertia:** sum of squared distance from each point (x_i) to its cluster (C_k)

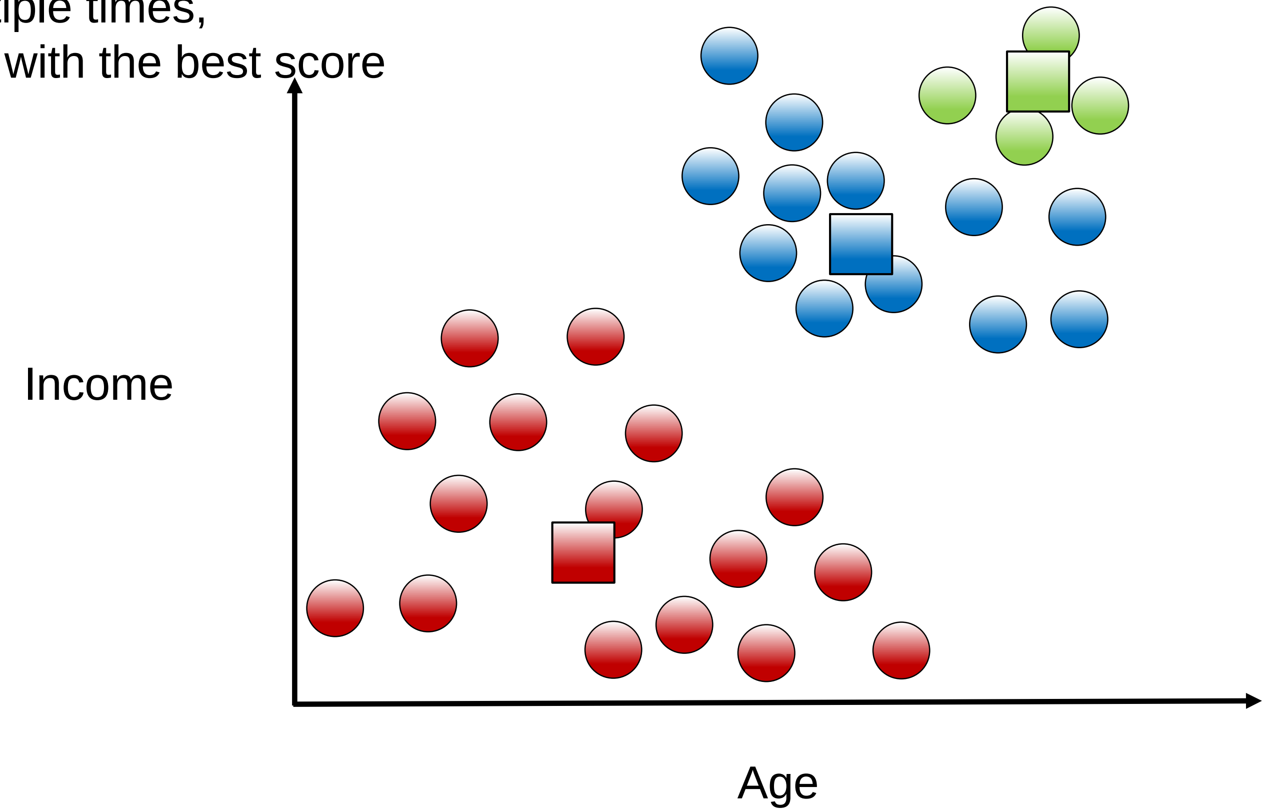
$$\sum_{i=1}^{n}(x_i - C_k)^2$$

- Smaller value corresponds to tighter clusters

- Other metrics can also be used

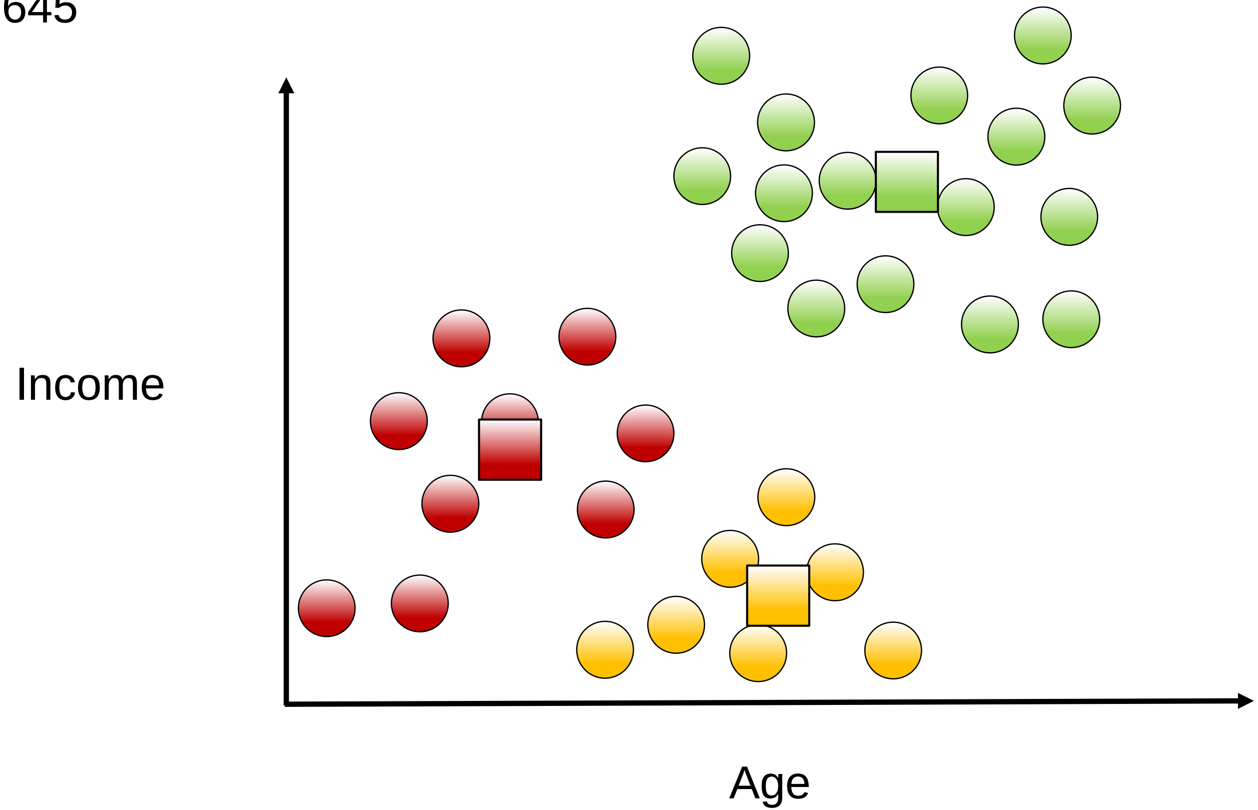
Which Model is the Right One?

Initiate multiple times,
take model with the best score



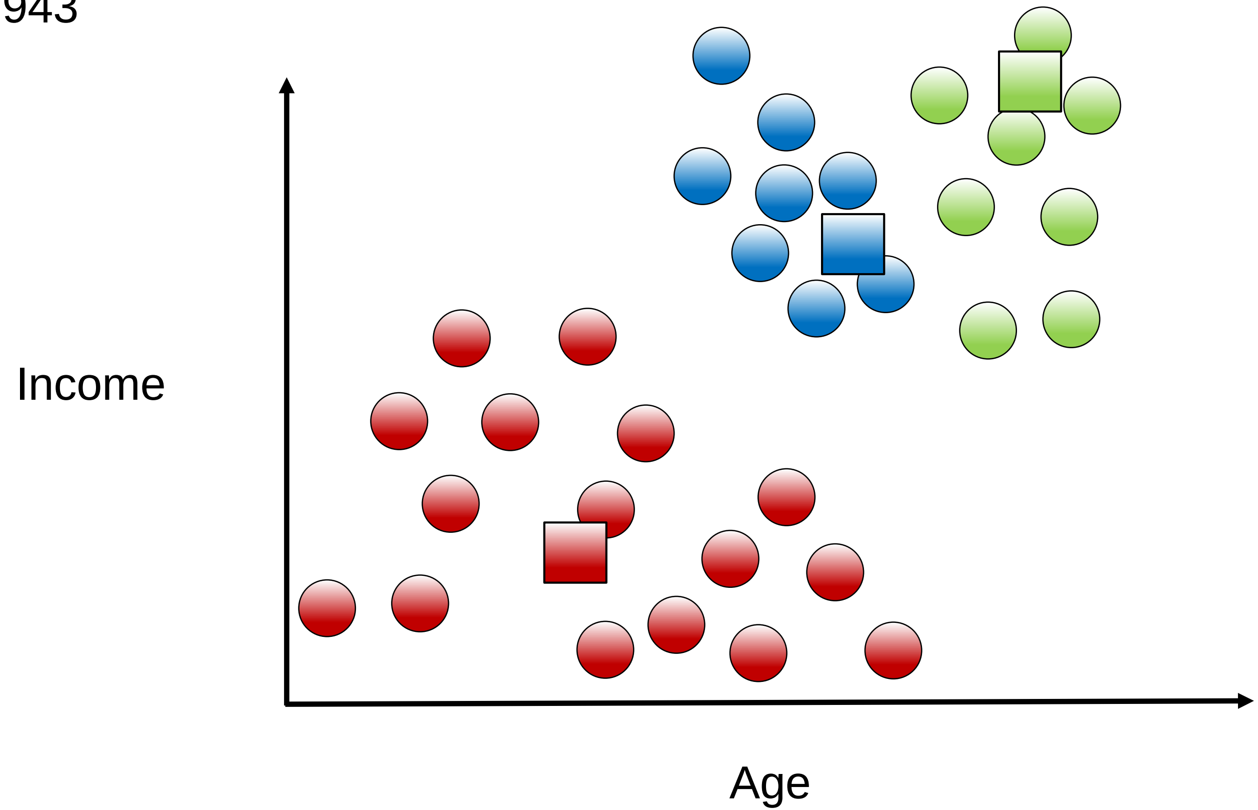
Which Model is the Right One?

Inertia = 12.645



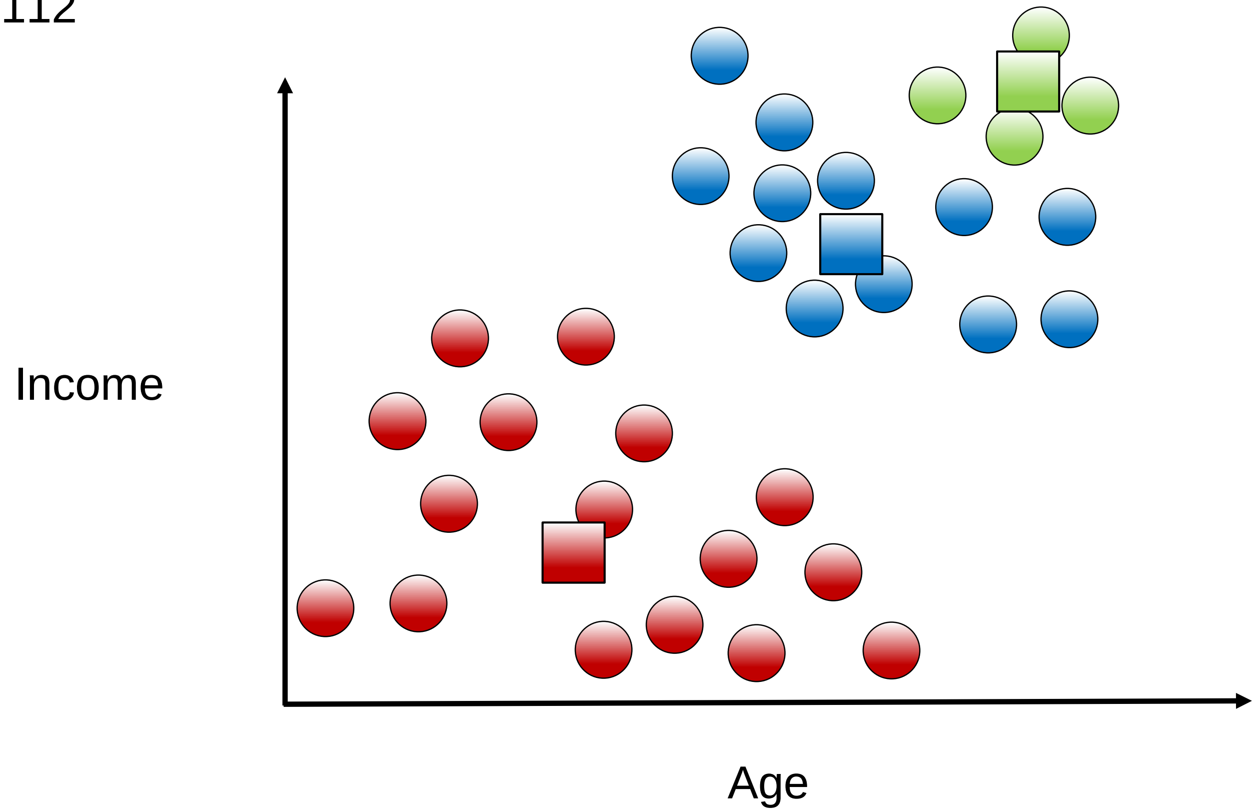
Which Model is the Right One?

Inertia = 12.943

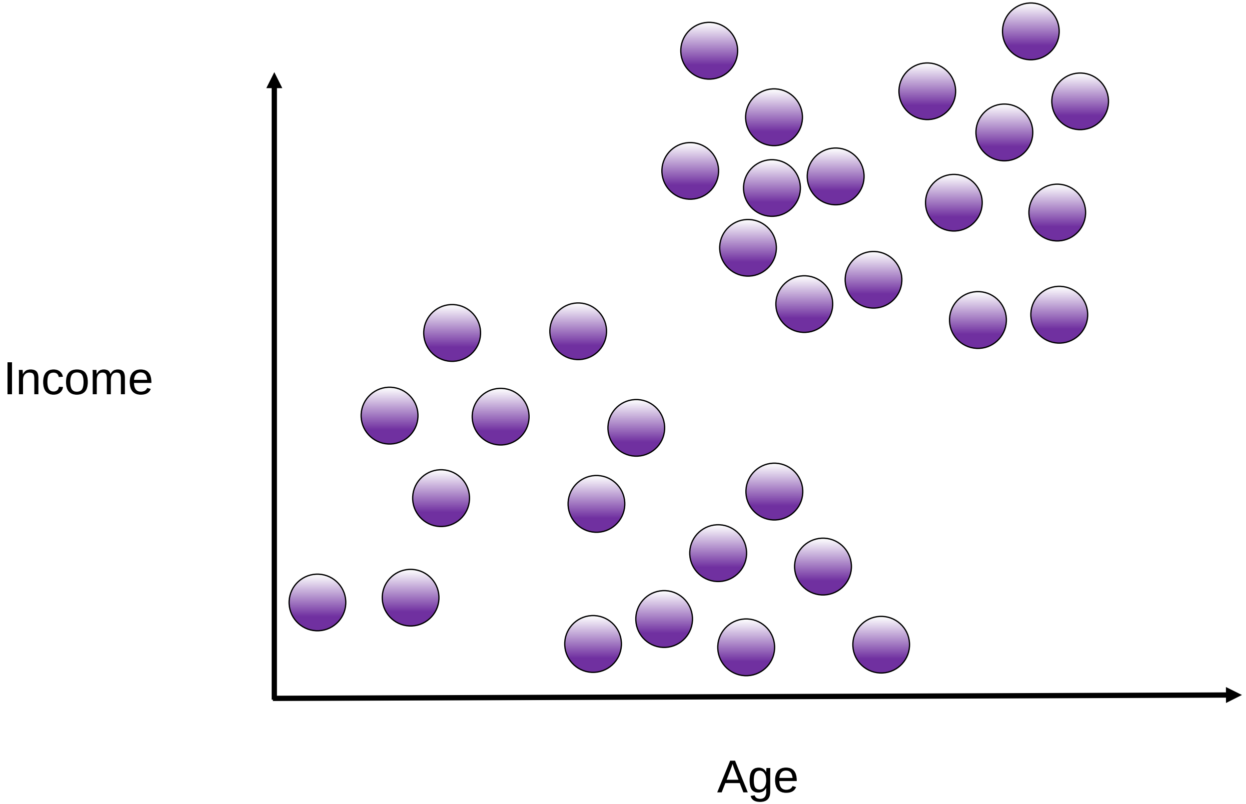


Which Model is the Right One?

Inertia = 13.112

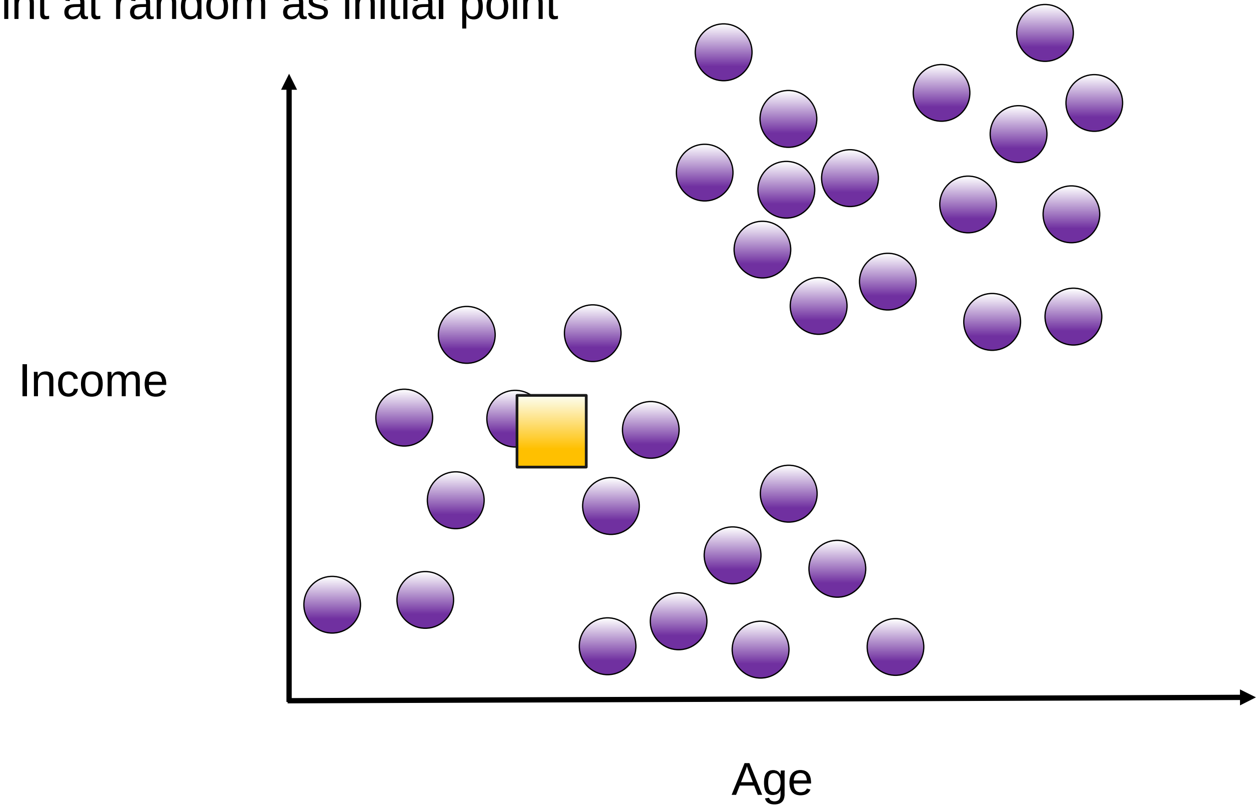


Smarter Initialization of K-Means Clusters



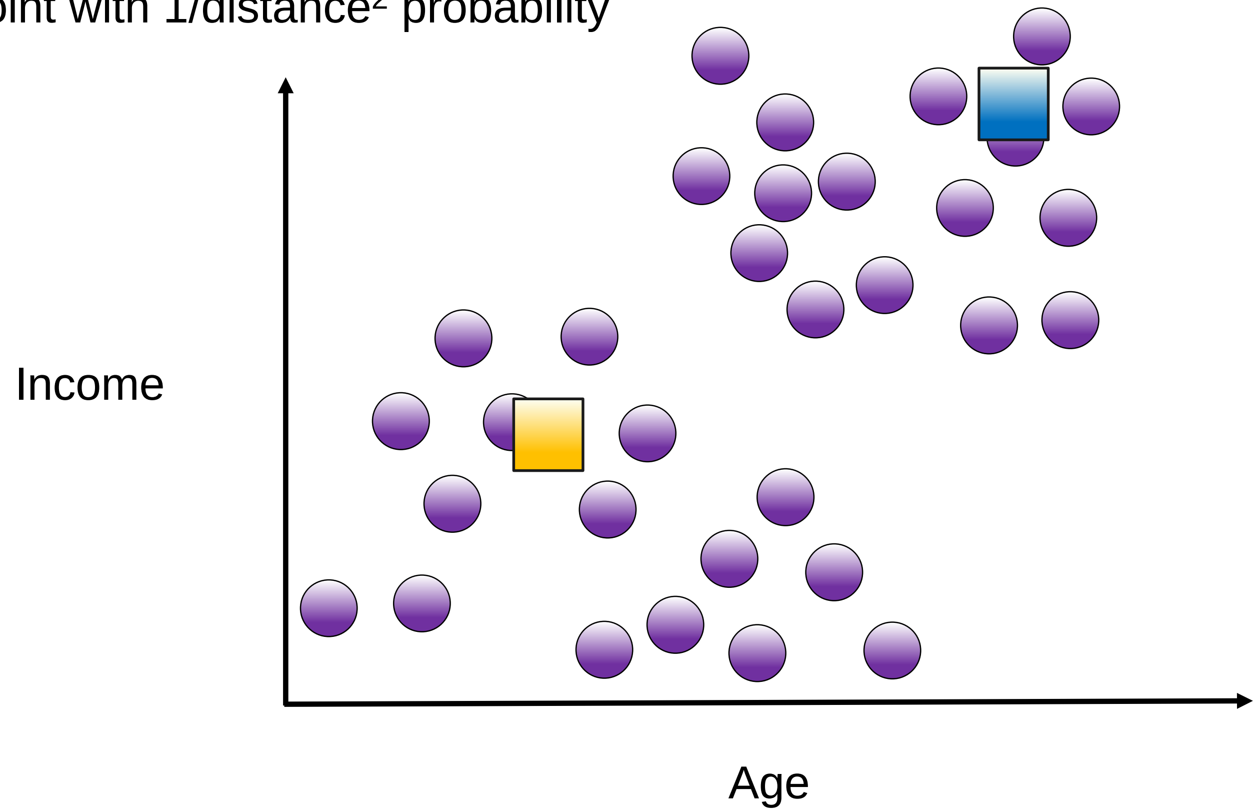
Smarter Initialization of K-Means Clusters

Pick one point at random as initial point



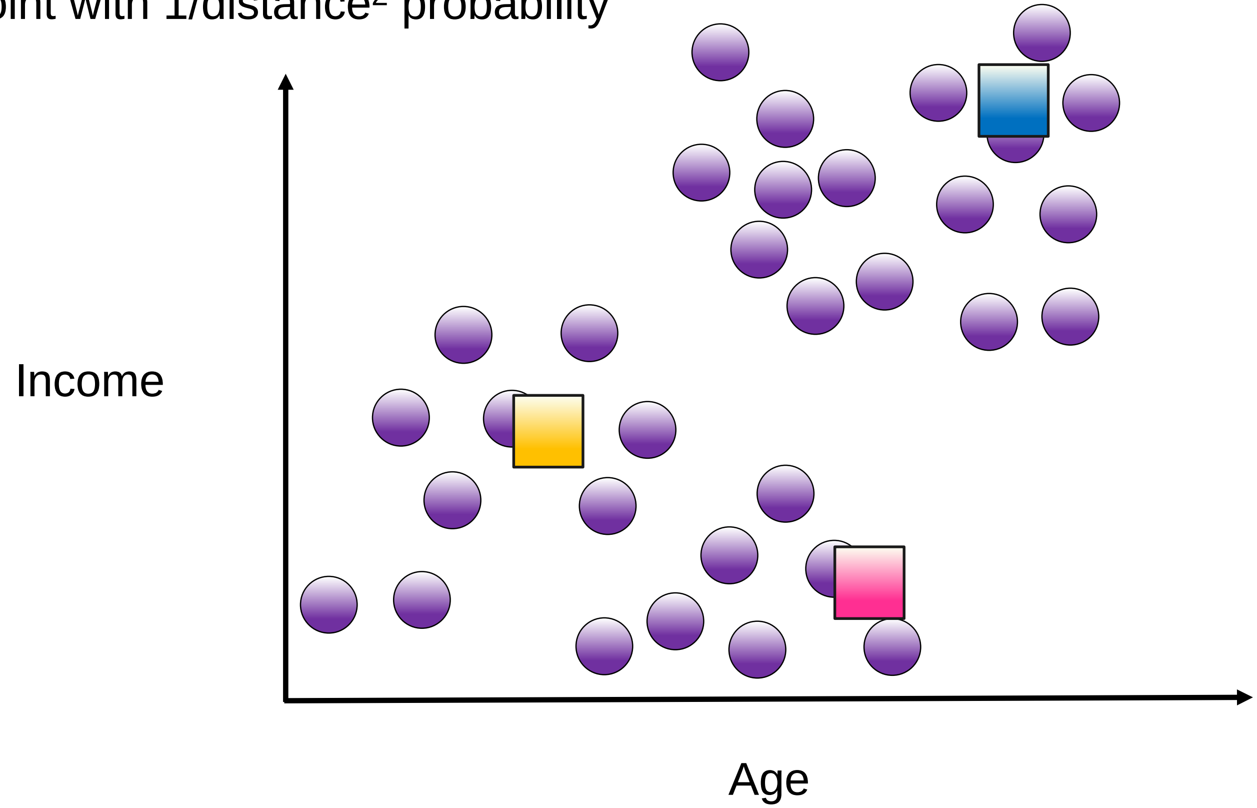
Smarter Initialization of K-Means Clusters

Pick next point with 1/distance^2 probability



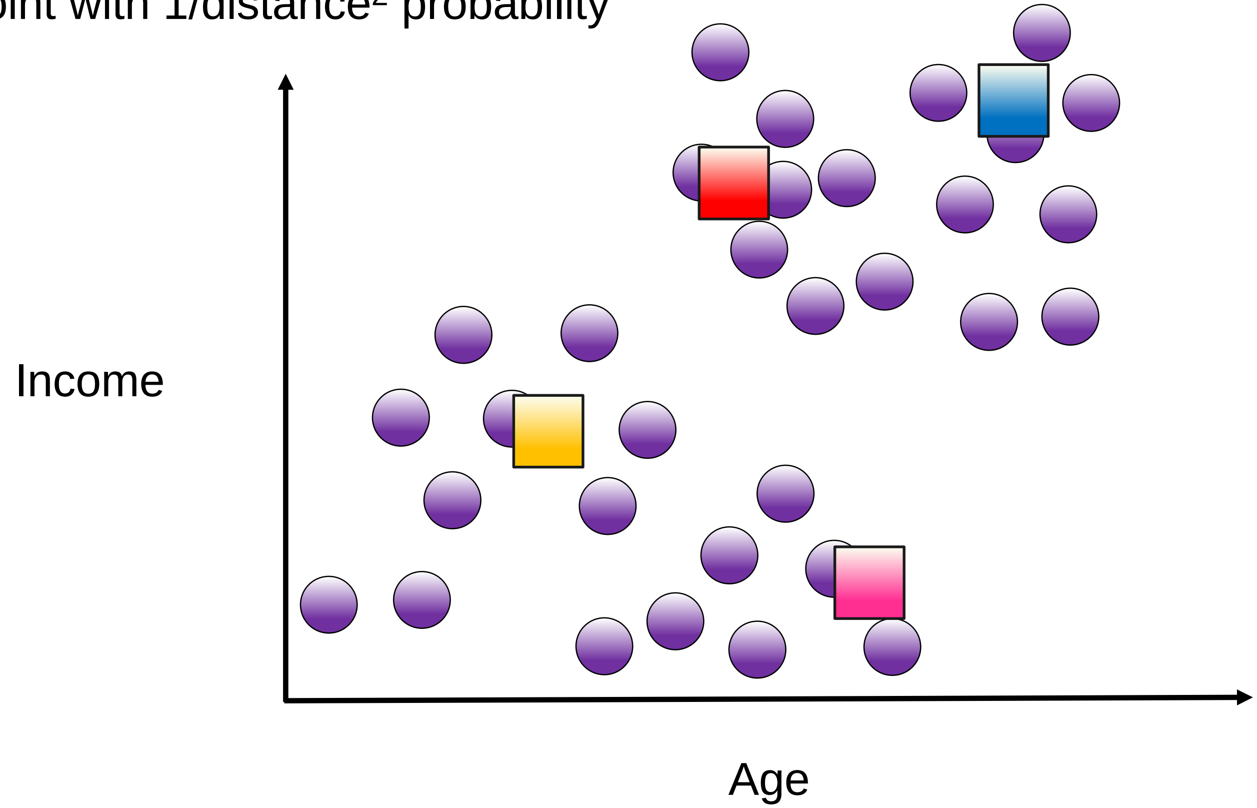
Smarter Initialization of K-Means Clusters

Pick next point with $1/\text{distance}^2$ probability



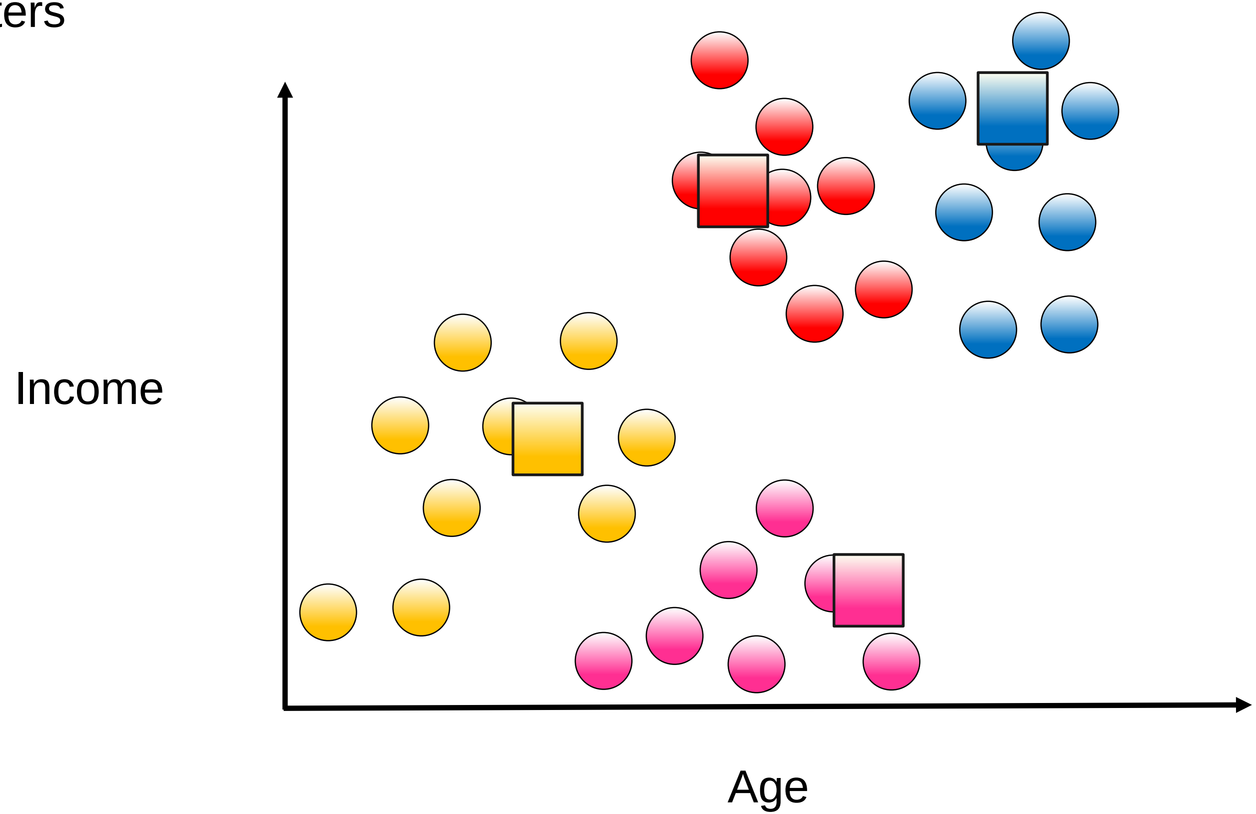
Smarter Initialization of K-Means Clusters

Pick next point with $1/\text{distance}^2$ probability



Smarter Initialization of K-Means Clusters

Assign clusters



Choosing the Right Number of Clusters

Choosing the Right Number of Clusters

- Sometimes the question has a K

Choosing the Right Number of Clusters

- Sometimes the question has a K
- Clustering similar jobs on 4 CPU cores (K=4)

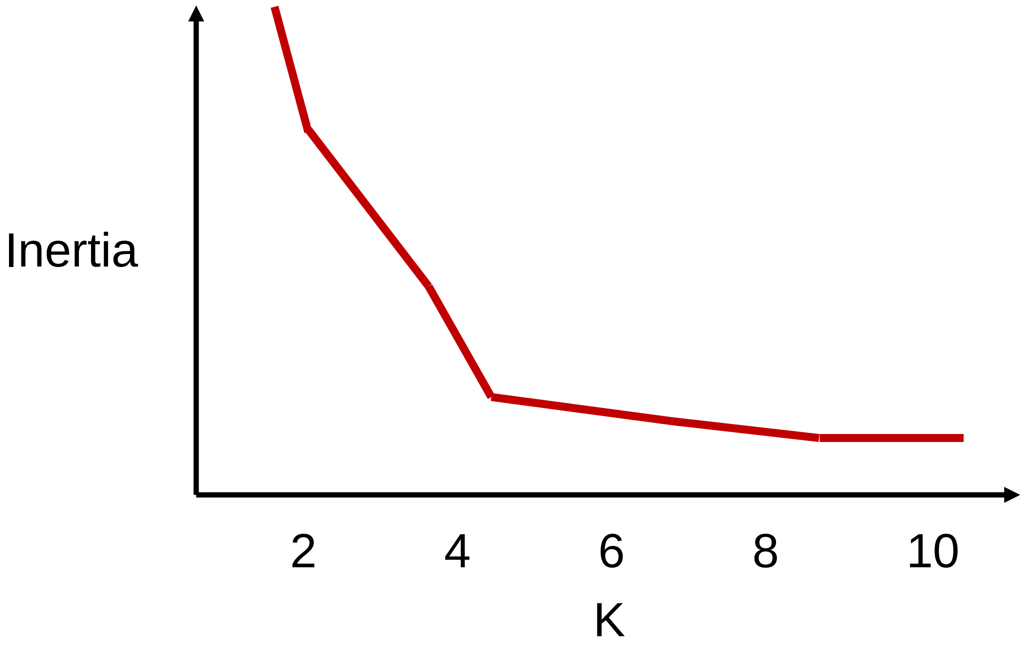
Choosing the Right Number of Clusters

- Sometimes the question has a K
- Clustering similar jobs on 4 CPU cores (K=4)
- A clothing design in 10 different sizes to cover most people (K=10)

Choosing the Right Number of Clusters

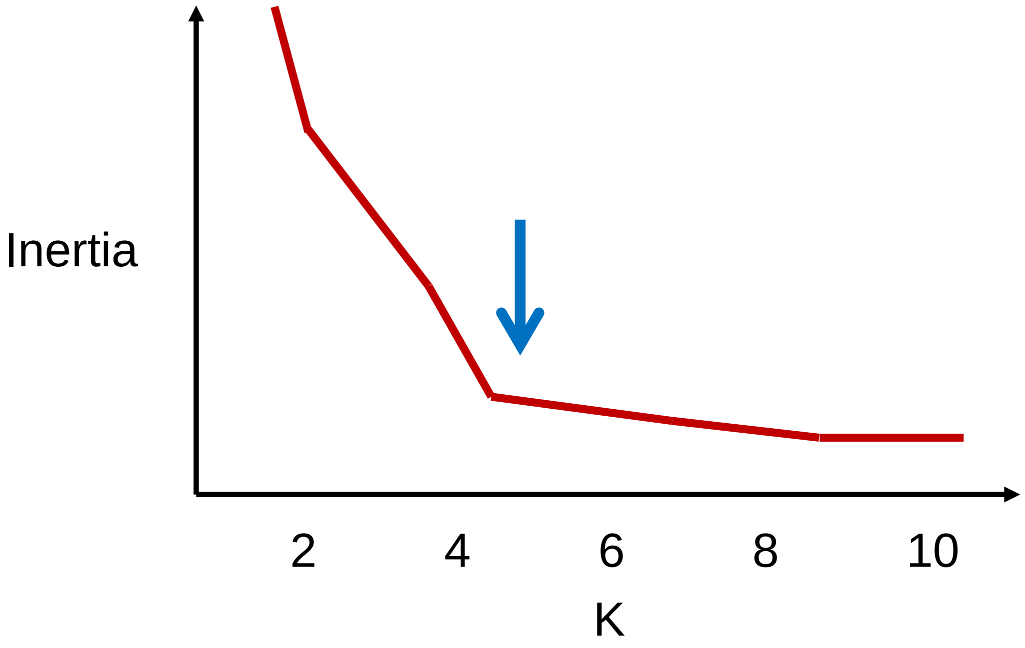
- Sometimes the question has a K
- Clustering similar jobs on 4 CPU cores (K=4)
- A clothing design in 10 different sizes to cover most people (K=10)
- A navigation interface for browsing scientific papers with 20 disciplines (K=20)

Choosing the Right Number of Clusters



- Inertia measures distance of point to cluster

Choosing the Right Number of Clusters



- Inertia measures distance of point to cluster
- Value decreases with increasing K as long as cluster density increases

K-Means: The Syntax

Import the class containing the clustering method

```
from sklearn.cluster import KMeans
```

K-Means: The Syntax

Import the class containing the clustering method

```
from sklearn.cluster import KMeans
```

Create an instance of the class

```
kmeans = KMeans(n_clusters=3,
                init='k-means++')
```

K-Means: The Syntax

Import the class containing the clustering method

```
from sklearn.cluster import KMeans
```

Create an instance of the class

```
kmeans = KMeans(n_clusters=3,
                init='k-means++')
```

final number of clusters

K-Means: The Syntax

Import the class containing the clustering method

```
from sklearn.cluster import KMeans
```

Create an instance of the class

```
kmeans = KMeans(n_clusters=3,
                init='k-means++')
```

kmeans++
cluster
initiation

K-Means: The Syntax

Import the class containing the clustering method

```
from sklearn.cluster import KMeans
```

Create an instance of the class

```
kmeans = KMeans(n_clusters=3,
                init='k-means++')
```

Fit the instance on the data and then predict clusters for new data

```
kmeans = kmeans.fit(X1)
```

K-Means: The Syntax

Import the class containing the clustering method

```
from sklearn.cluster import KMeans
```

Create an instance of the class

```
kmeans = KMeans(n_clusters=3,
                init='k-means++')
```

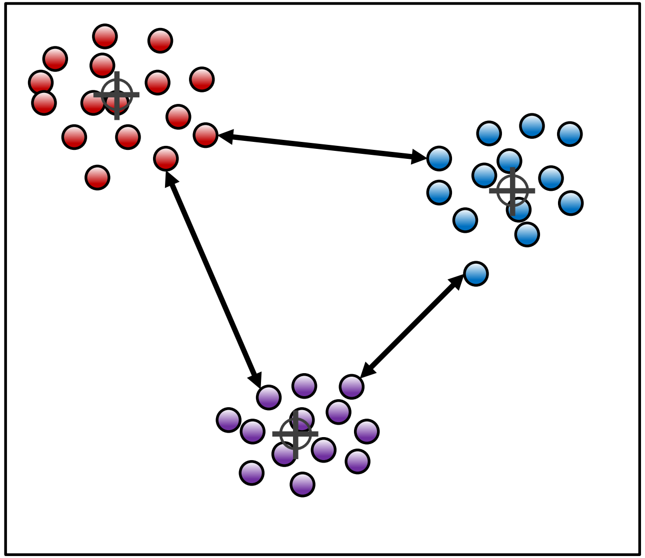
Fit the instance on the data and then predict clusters for new data

```
kmeans = kmeans.fit(X1)
y_predict = kmeans.predict(X2)
```

Can also be used in batch mode with MiniBatchKMeans.

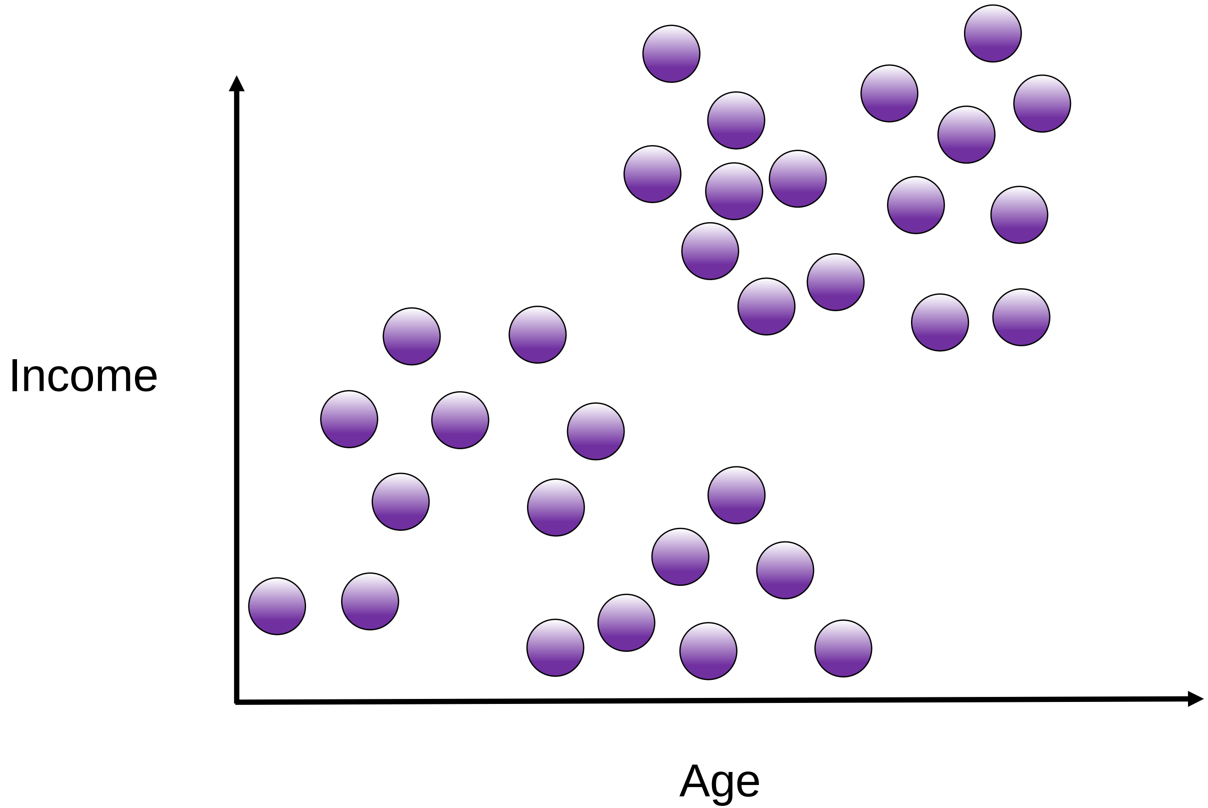
Distance Metrics

Distance Metric Choice

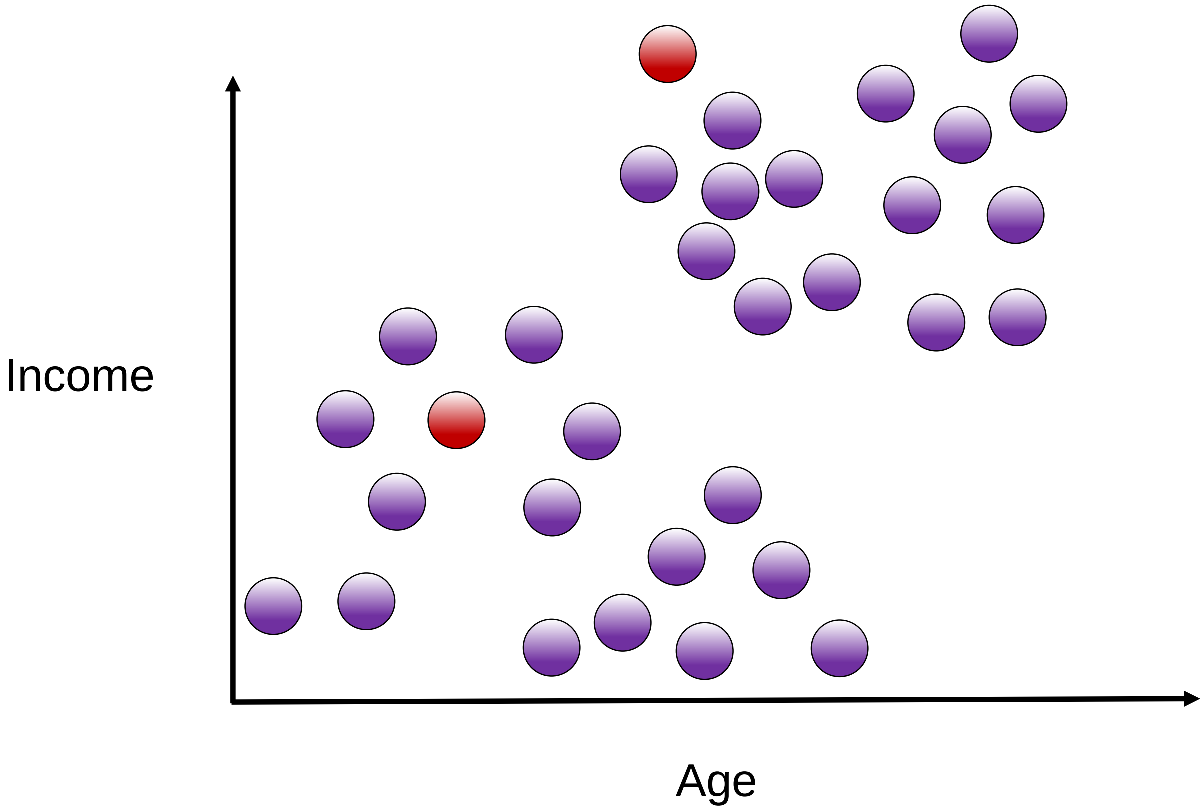


- Choice of distance metric is extremely important to clustering success
- Each metric has strengths and most appropriate use-cases…
- …but sometimes choice of distance metric is also based on empirical evaluation

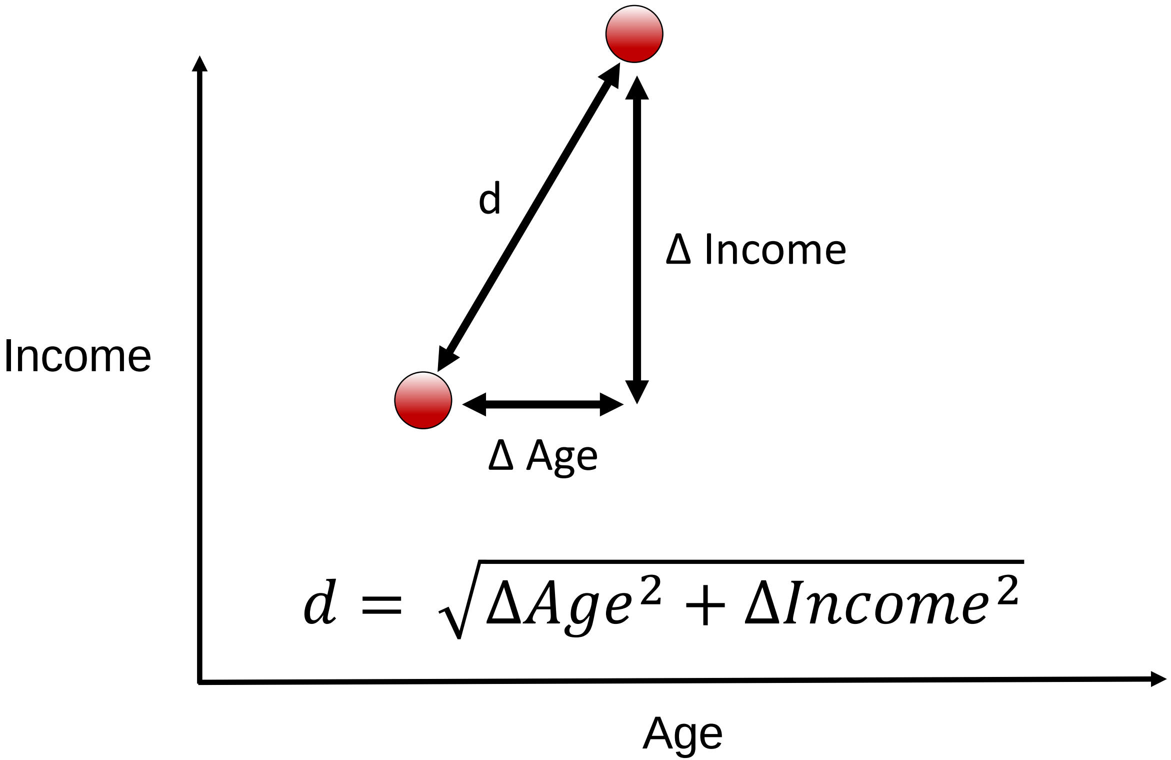
Euclidean Distance



Euclidean Distance

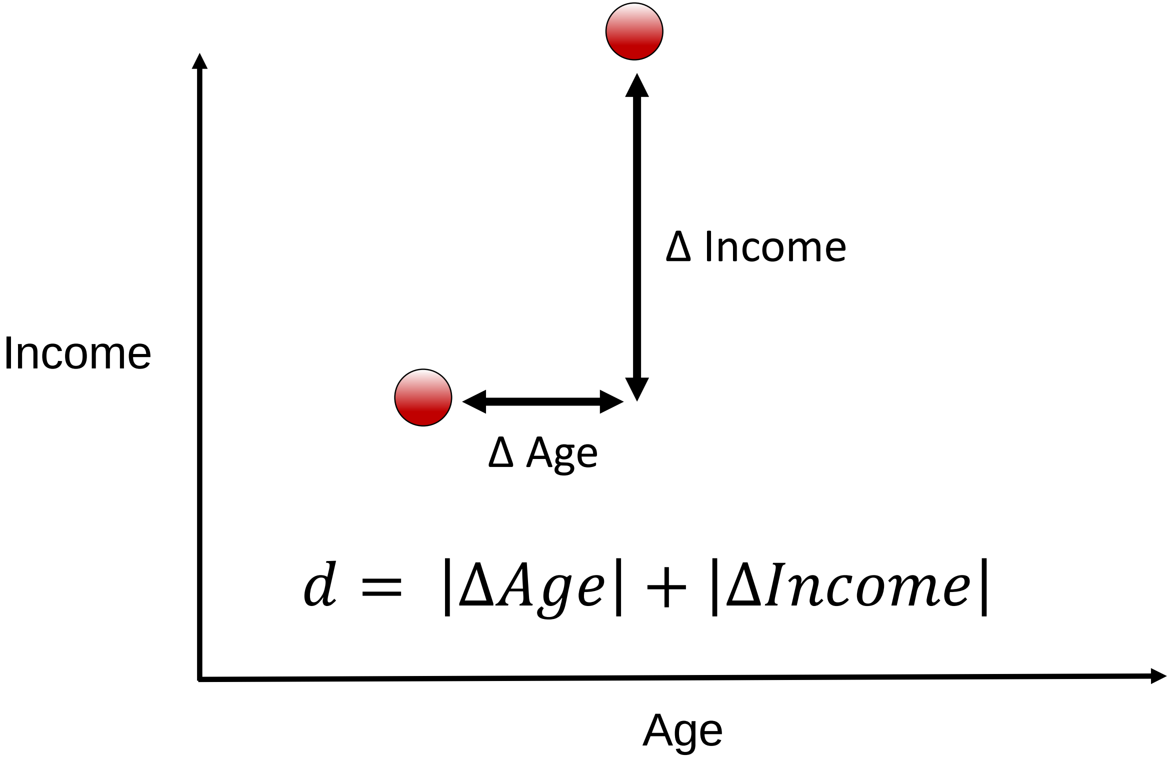


Euclidean Distance (L2 Distance)



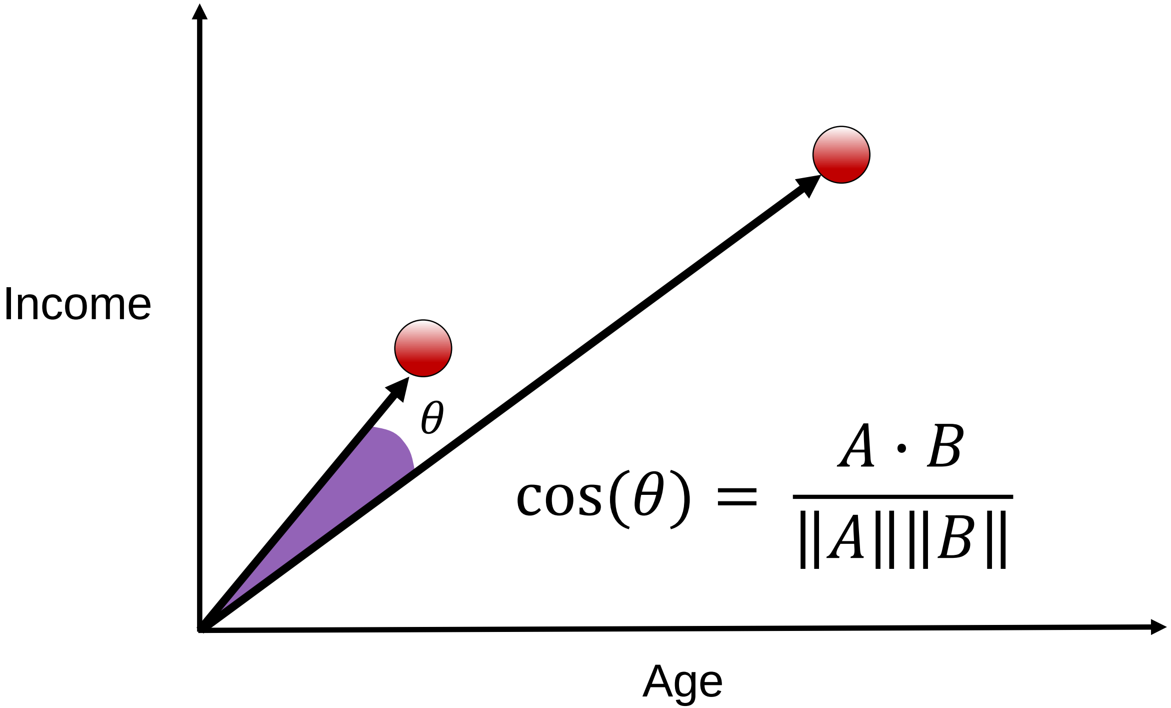
$$d = \sqrt{\Delta Age^2 + \Delta Income^2}$$

Manhattan Distance (L1 or City Block Distance)



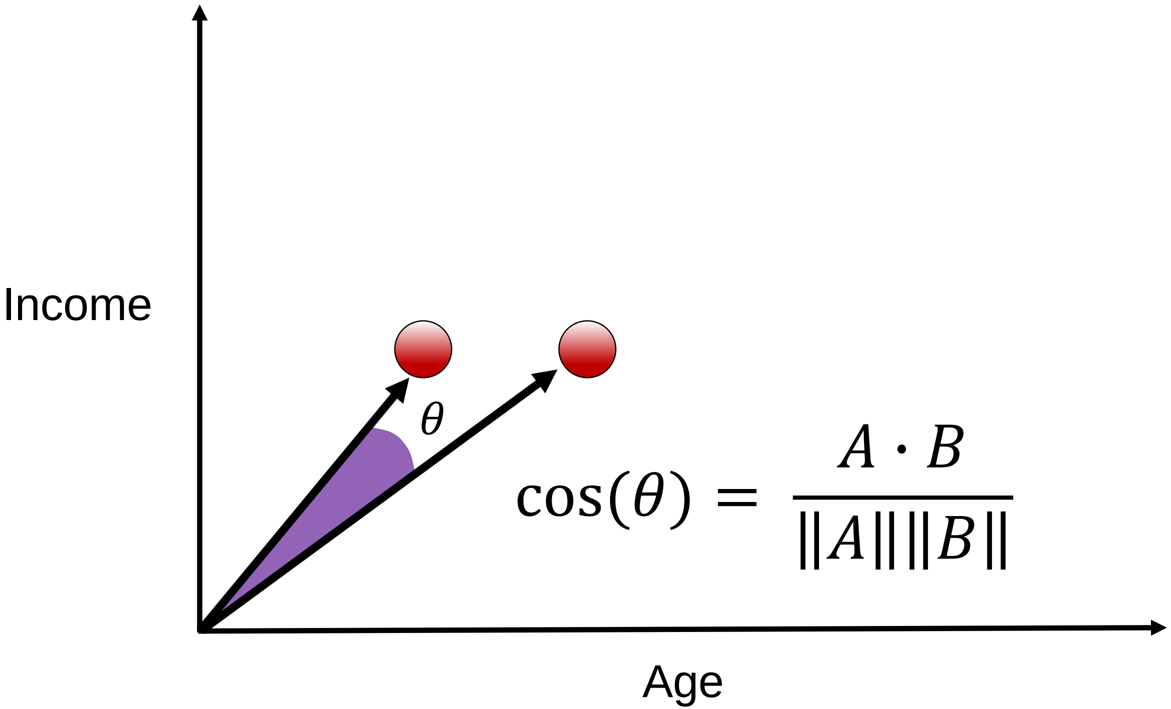
$$d = |\Delta Age| + |\Delta Income|$$

Cosine Distance



$$\cos(\theta) = \frac{A \cdot B}{\|A\| \|B\|}$$

Cosine Distance



$$\cos(\theta) = \frac{A \cdot B}{\|A\| \|B\|}$$

Euclidean vs Cosine Distance

- Euclidean is useful for coordinate based measurements

Euclidean vs Cosine Distance

- Euclidean is useful for coordinate based measurements
- Cosine is better for data such as text where location of occurrence is less important

Euclidean vs Cosine Distance

- Euclidean is useful for coordinate based measurements
- Cosine is better for data such as text where location of occurrence is less important
- Euclidean distance is more sensitive to curse of dimensionality (see lesson 12)

Jaccard Distance

Applies to sets (like word occurrence)

- **Sentence A:** “I like chocolate ice cream.”
- set A = {I, like, chocolate, ice, cream}
- **Sentence B:** “Do I want chocolate cream or vanilla cream?”
- set B = {Do, I, want, chocolate, cream, or, vanilla}

$$1 - \frac{A \cap B}{A \cup B} = 1 - \frac{len(shared)}{len(unique)}$$

Jaccard Distance

Applies to sets (like word occurrence)

- **Sentence A:** “I like chocolate ice cream.”
- set A = {**I**, like, **chocolate**, ice, **cream**}
- **Sentence B:** “Do I want chocolate cream or vanilla cream?”
- set B = {Do, **I**, want, **chocolate**, **cream**, or, vanilla}

$$1 - \frac{A \cap B}{A \cup B} = 1 - \frac{3}{9}$$

Distance Metrics: The Syntax

Import the general pairwise distance function

```
from sklearn.metrics import pairwise_distances
```

Distance Metrics: The Syntax

Import the general pairwise distance function

```
from sklearn.metrics import pairwise_distances
```

Calculate the distances

```
dist = pairwise_distances(X, Y,
                          metric='euclidean')
```

Distance Metrics: The Syntax

Import the general pairwise distance function

```
from sklearn.metrics import pairwise_distances
```

Calculate the distances

```
dist = pairwise_distances(X, Y,
                          metric='euclidean')
```

distance metric choice

Distance Metrics: The Syntax

Import the general pairwise distance function

```
from sklearn.metrics import pairwise_distances
```

Calculate the distances

```
dist = pairwise_distances(X, Y,
                          metric='euclidean')
```

Other distance metric choices are: cosine, manhattan, jaccard, etc.

Distance Metrics: The Syntax

Import the general pairwise distance function

```
from sklearn.metrics import pairwise_distances
```

Calculate the distances

```
dist = pairwise_distances(X, Y,
                          metric='euclidean')
```

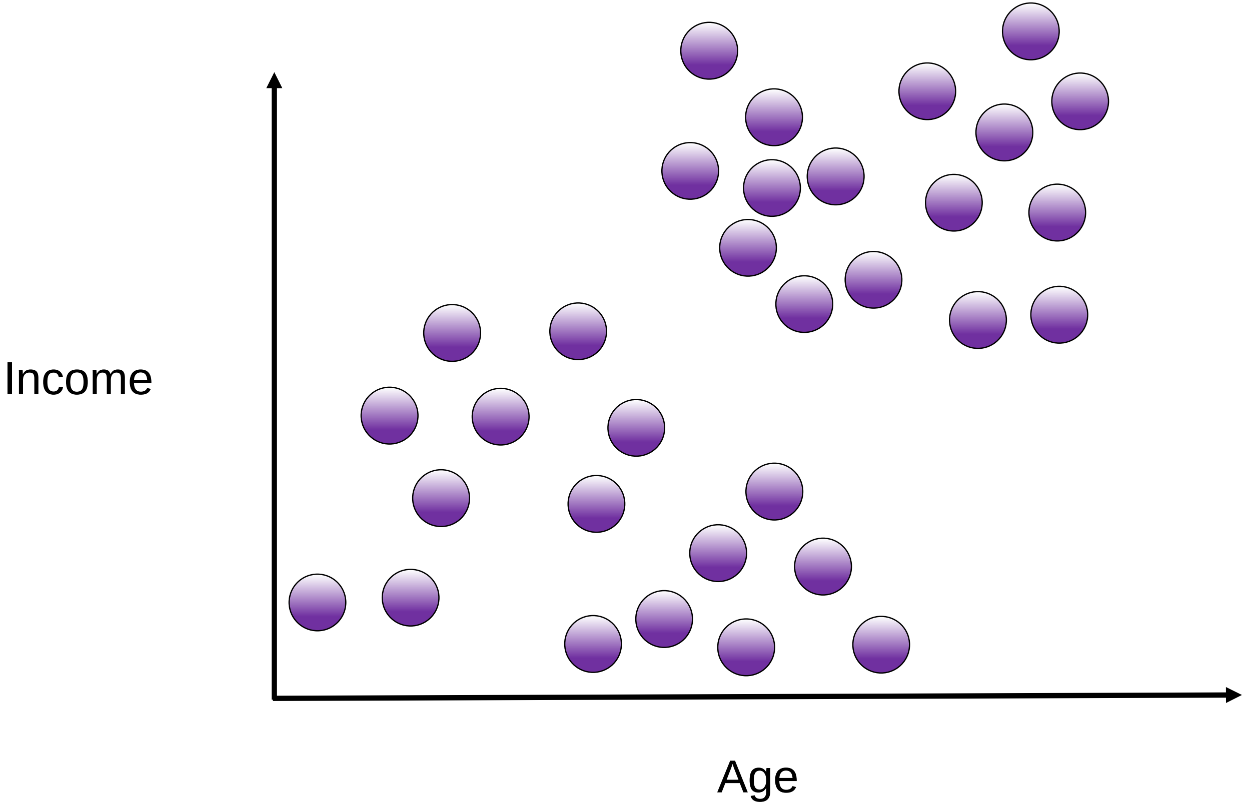
Other distance metric choices are: cosine, manhattan, jaccard, etc.

Distance metric methods can also be imported specifically, e.g.:

```
from sklearn.metrics import euclidean_distances
```

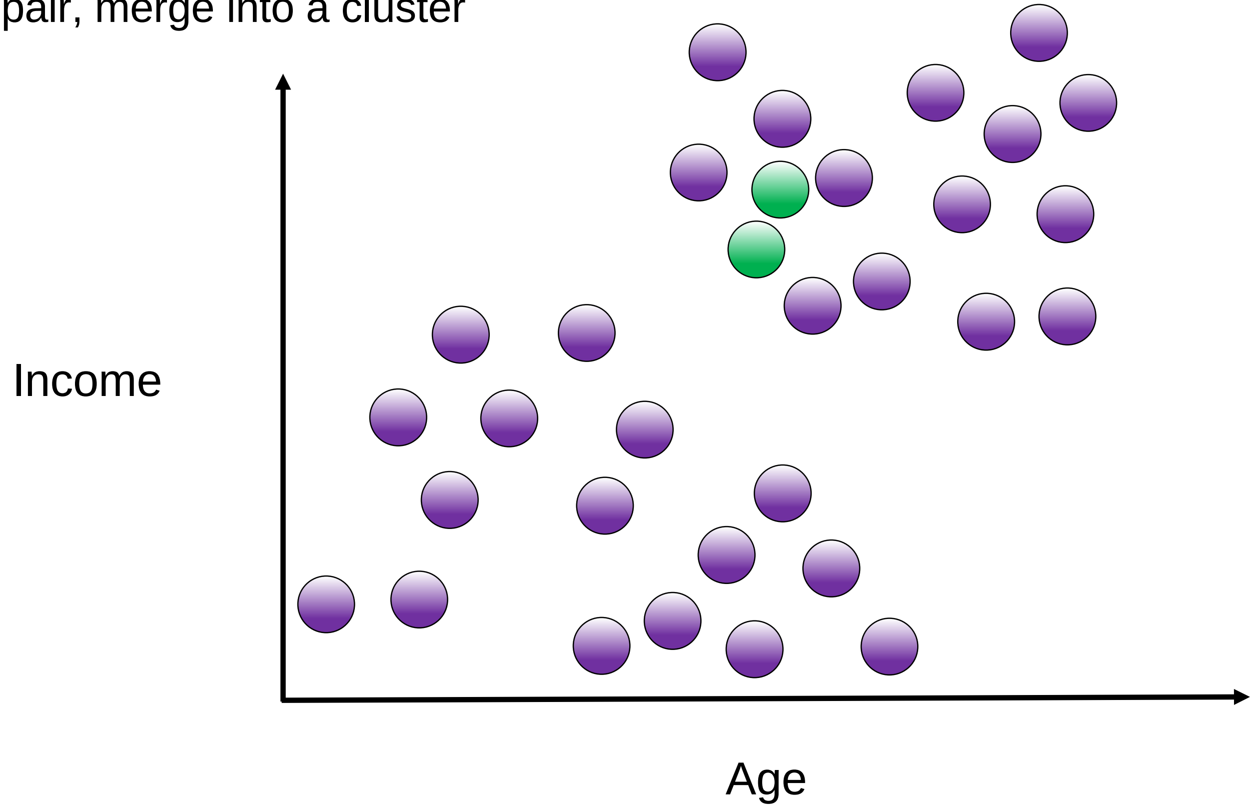
Hierarchical Agglomerative Clustering

Hierarchical Agglomerative Clustering



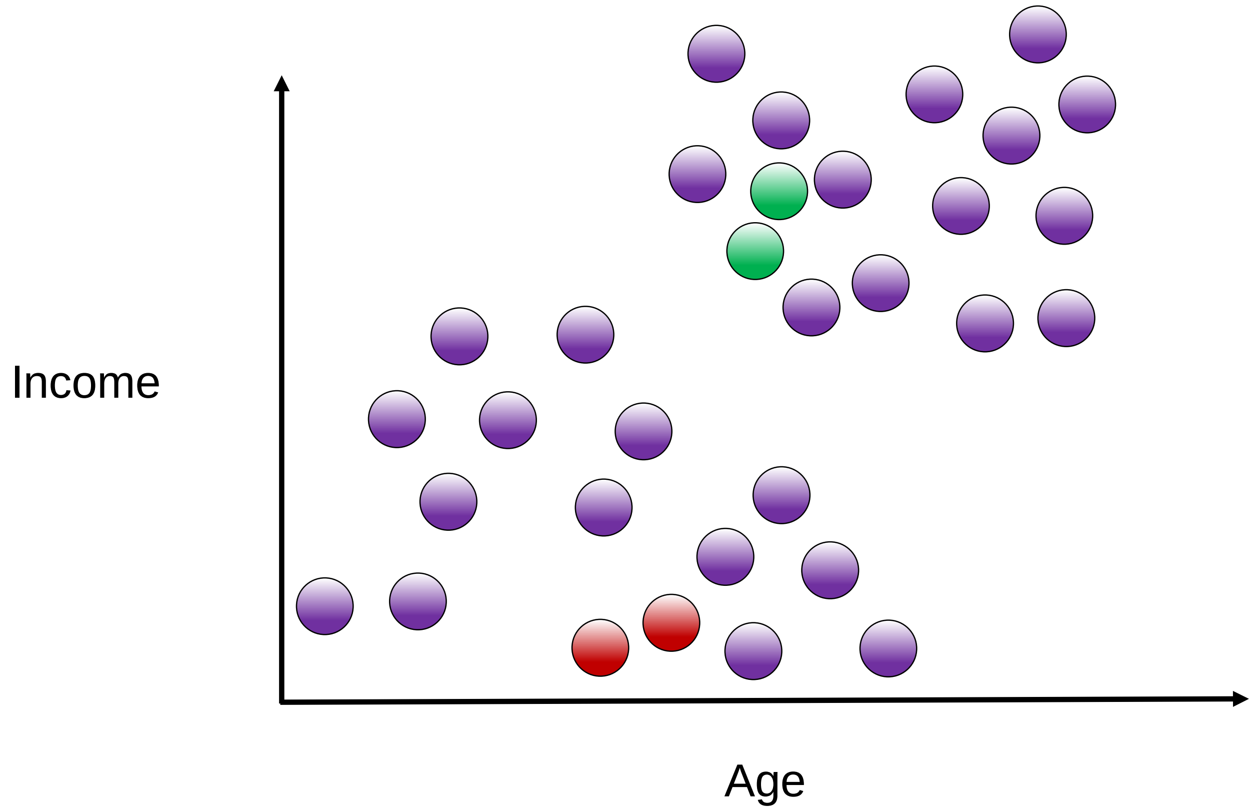
Hierarchical Agglomerative Clustering

Find closest pair, merge into a cluster



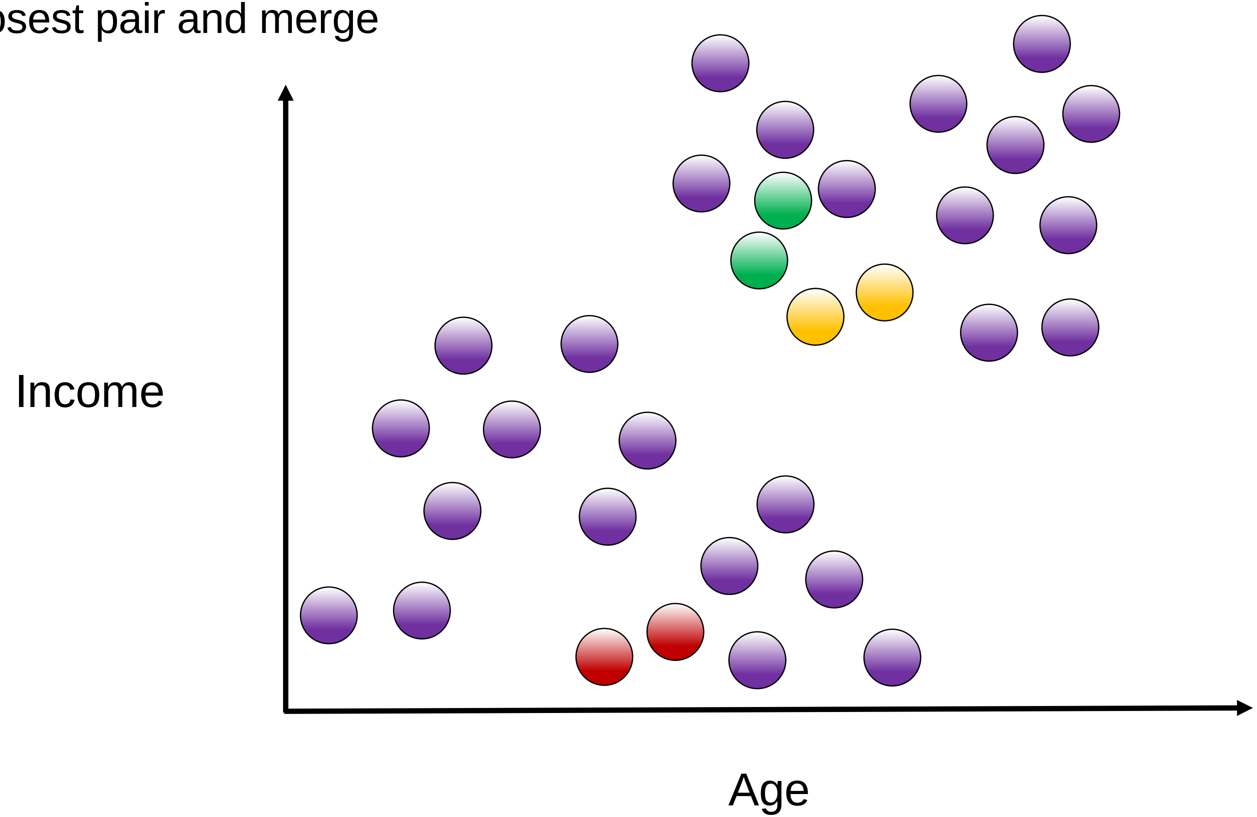
Hierarchical Agglomerative Clustering

Find next closest pair and merge



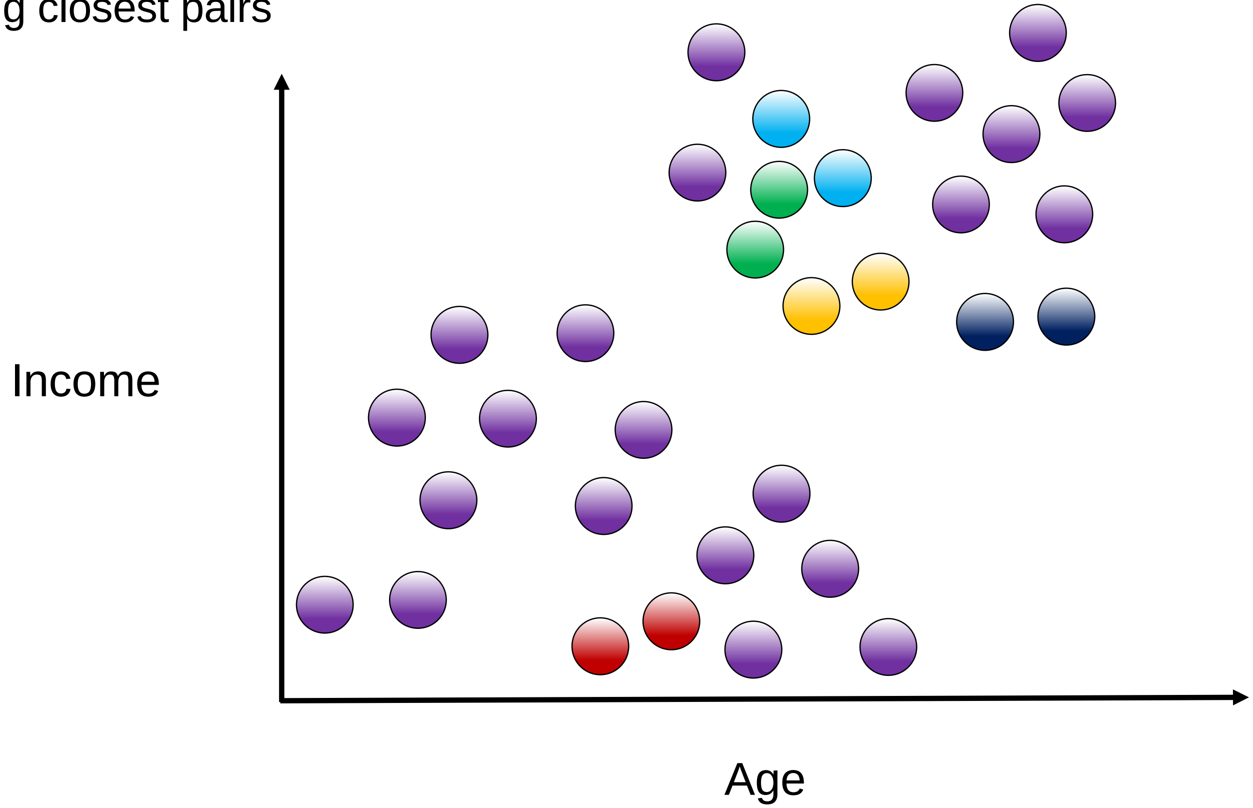
Hierarchical Agglomerative Clustering

Find next closest pair and merge



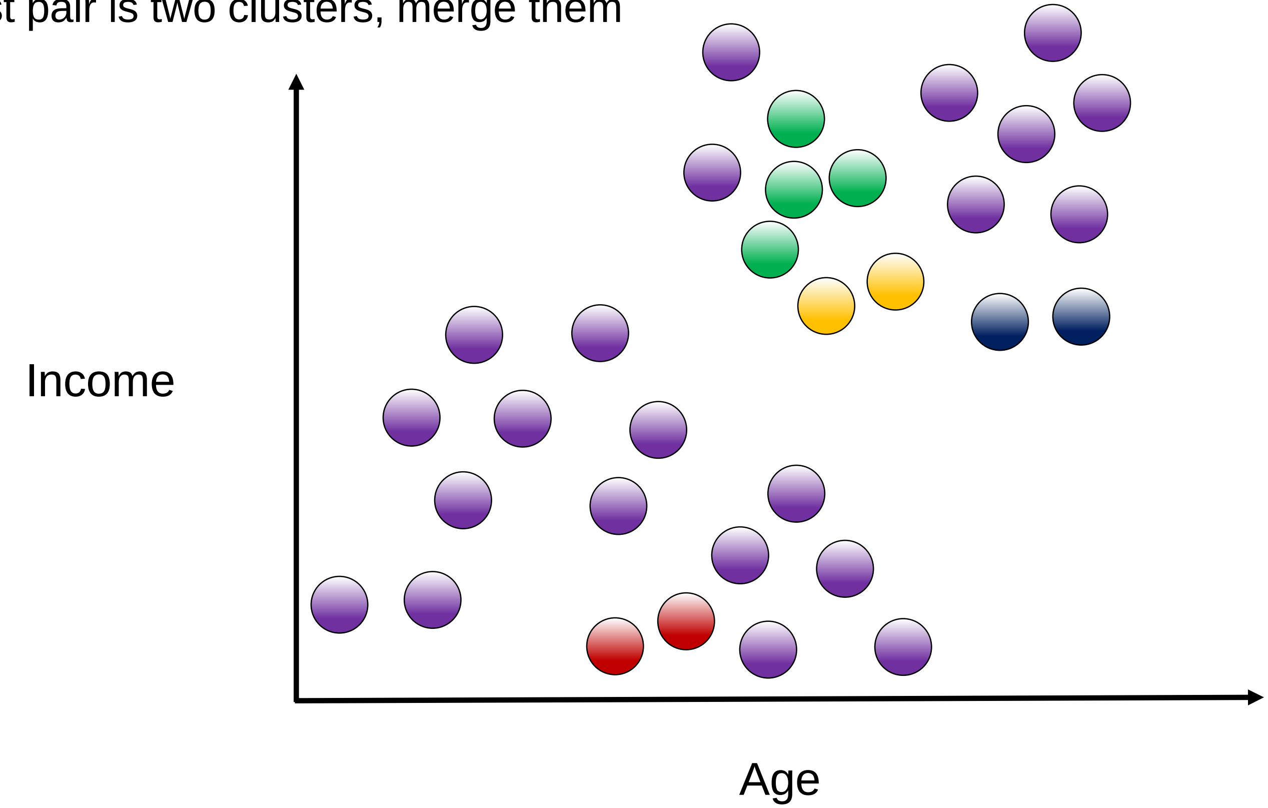
Hierarchical Agglomerative Clustering

Keep merging closest pairs



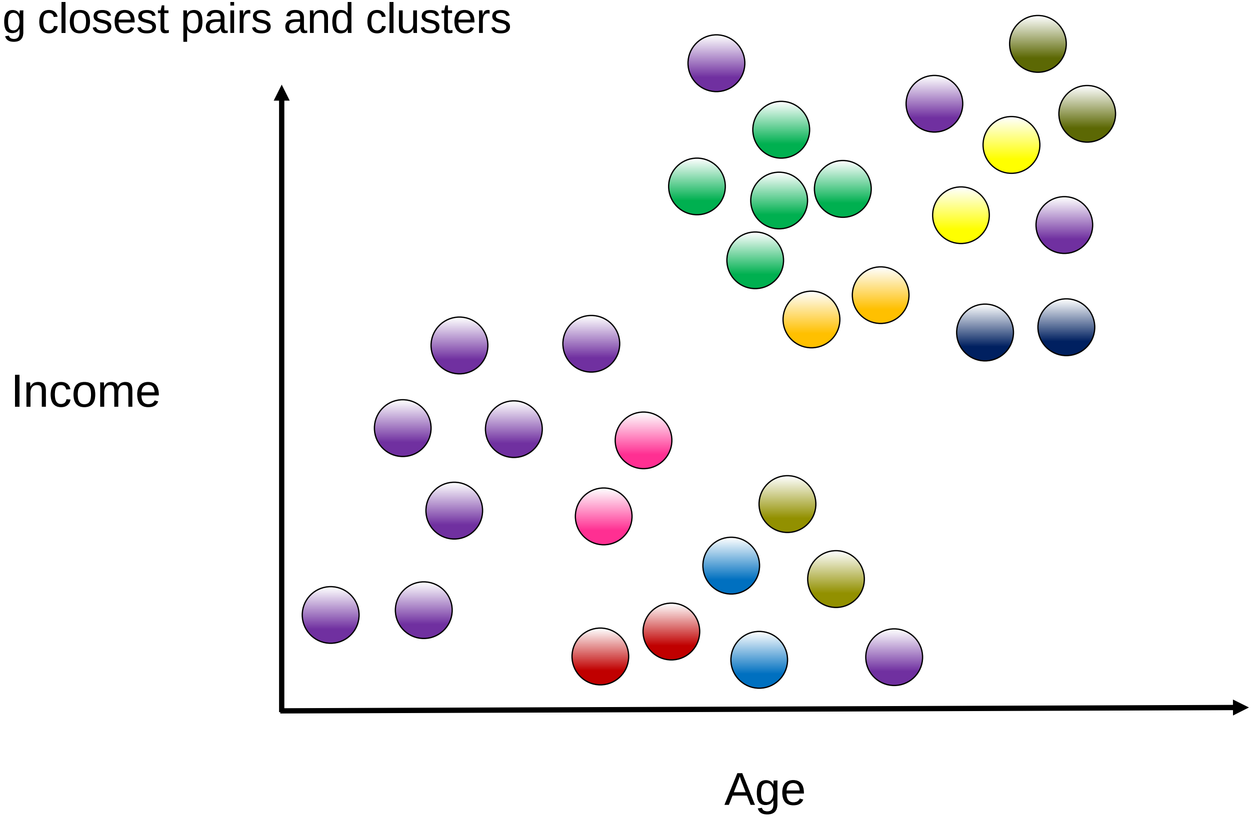
Hierarchical Agglomerative Clustering

If the closest pair is two clusters, merge them



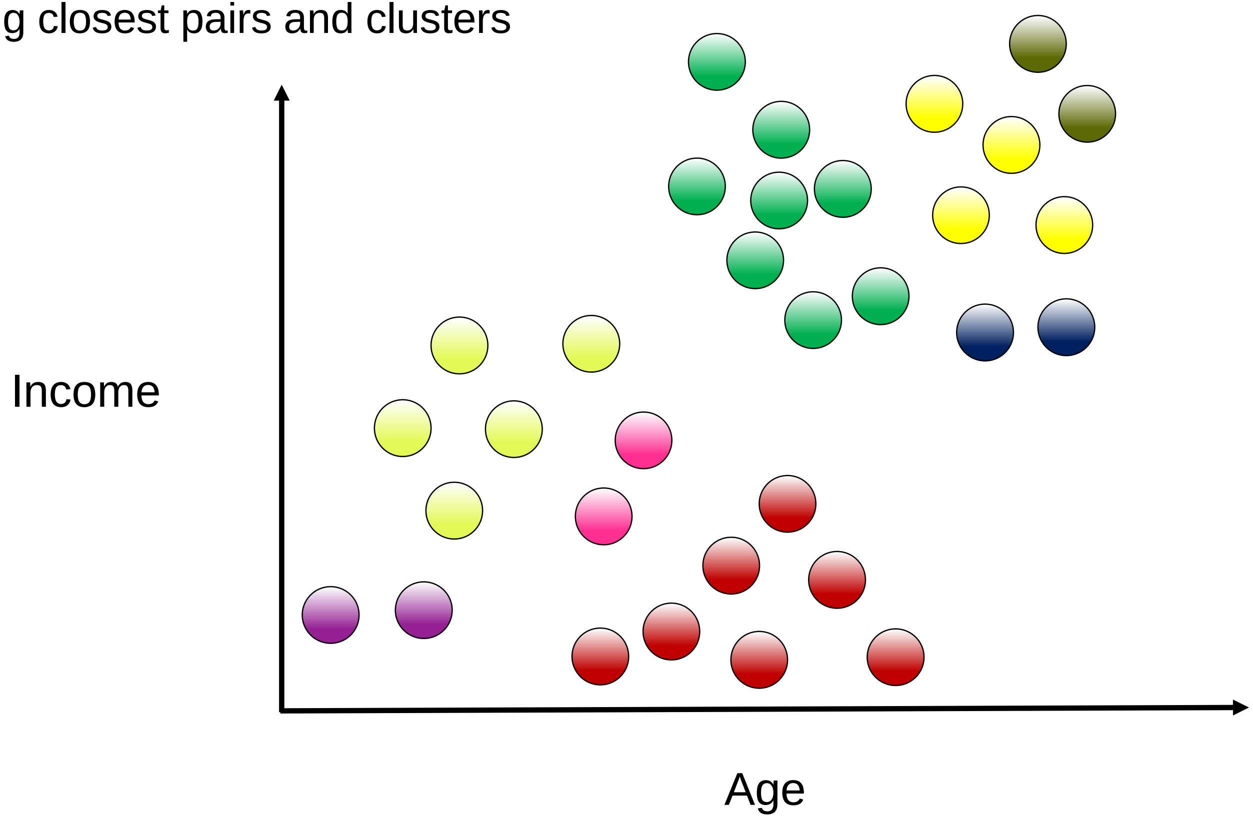
Hierarchical Agglomerative Clustering

Keep merging closest pairs and clusters



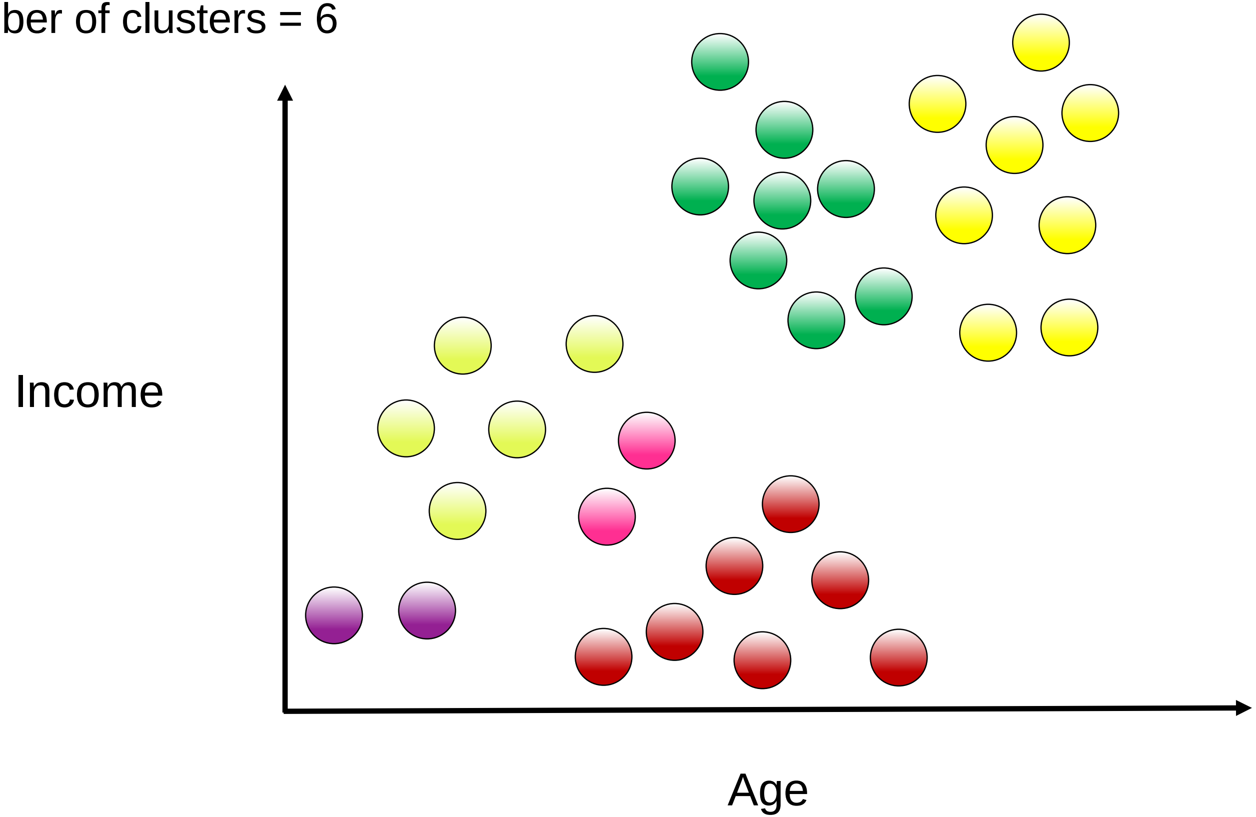
Hierarchical Agglomerative Clustering

Keep merging closest pairs and clusters



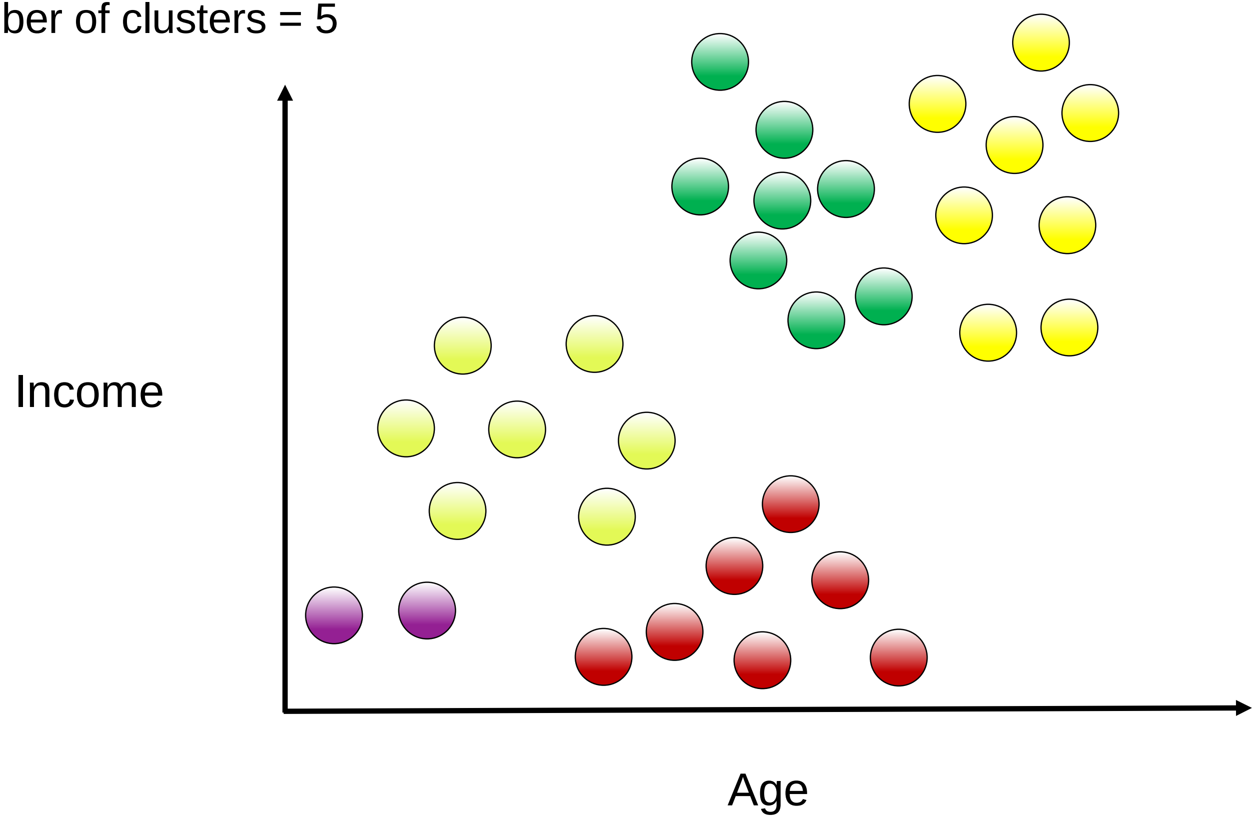
Hierarchical Agglomerative Clustering

Current number of clusters = 6



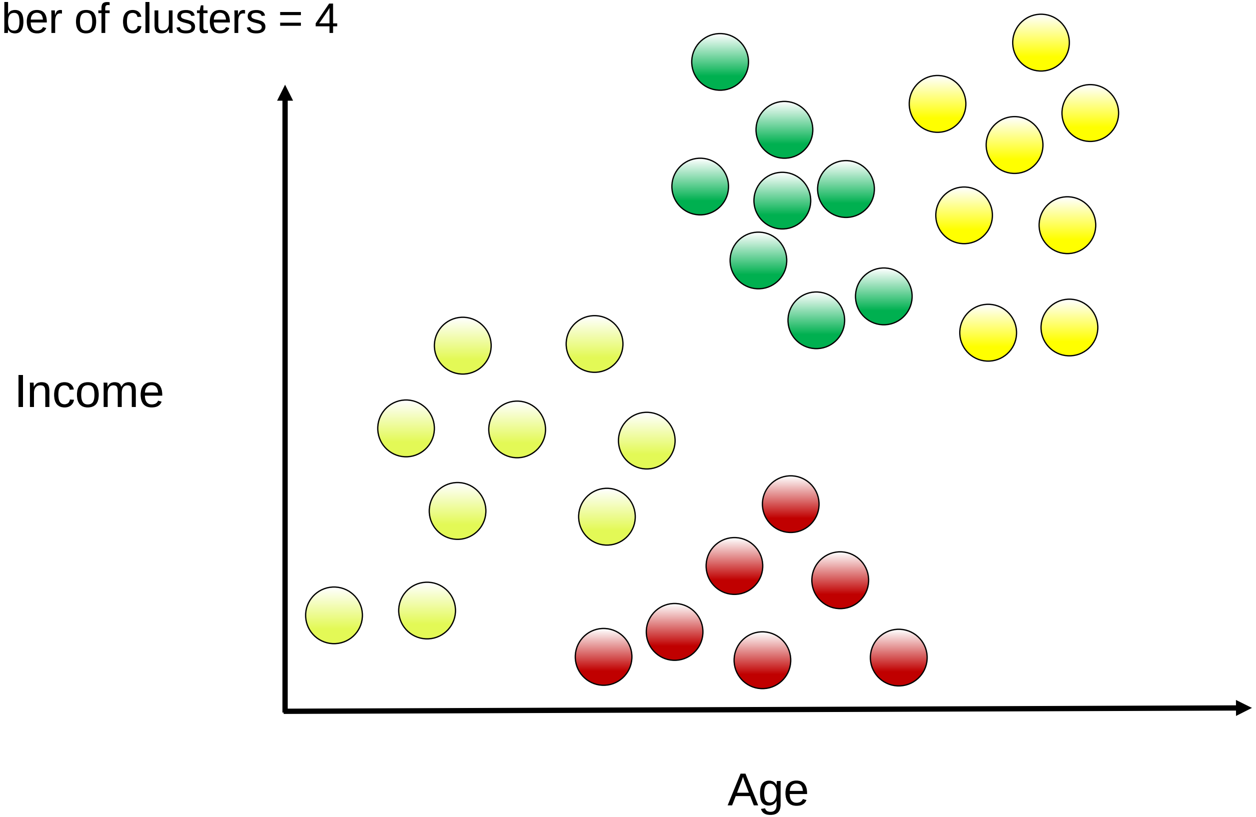
Hierarchical Agglomerative Clustering

Current number of clusters = 5



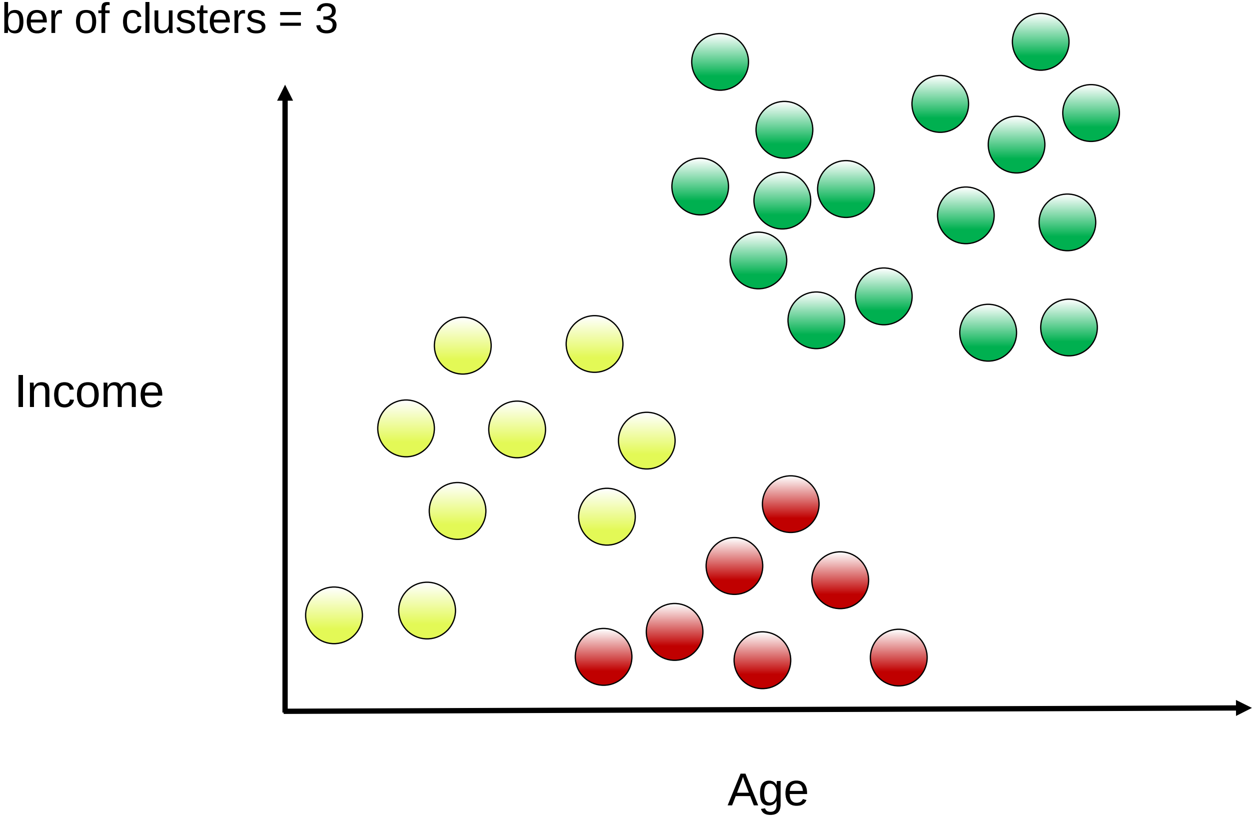
Hierarchical Agglomerative Clustering

Current number of clusters = 4



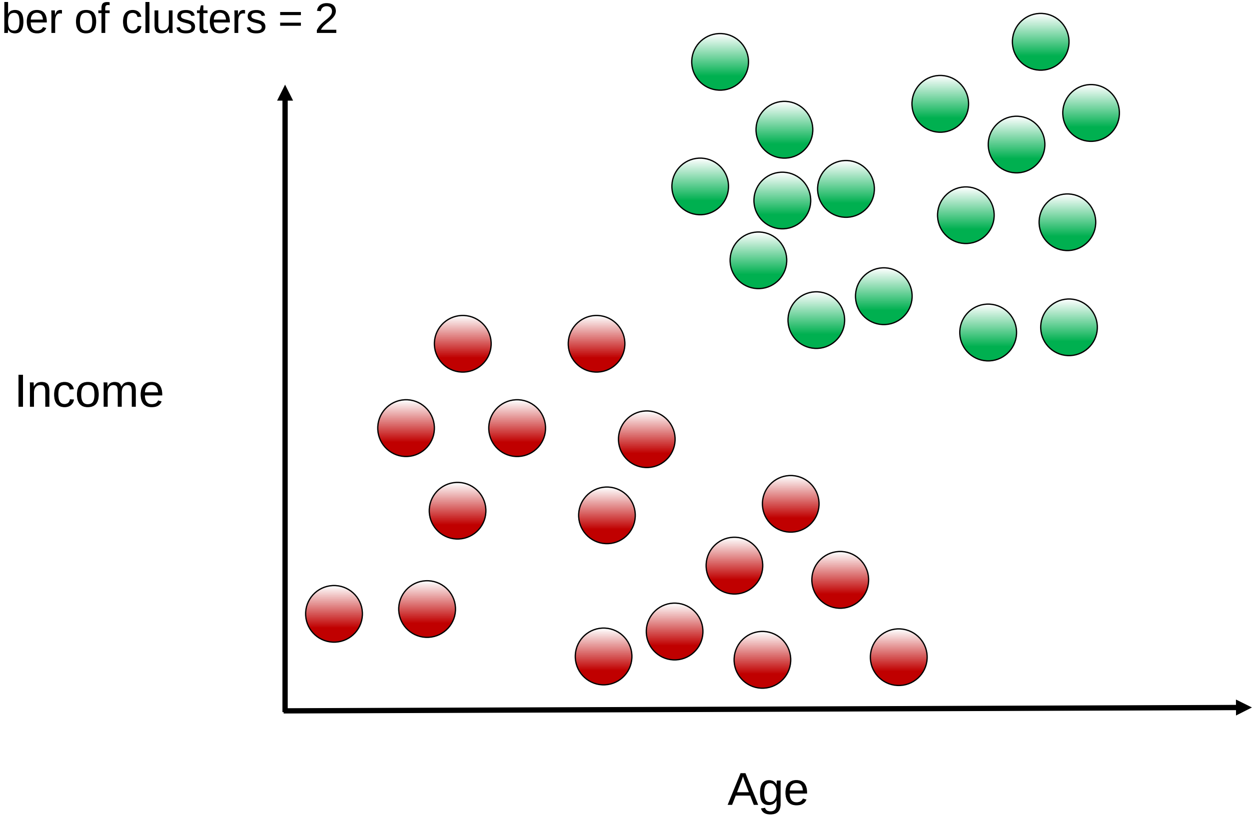
Hierarchical Agglomerative Clustering

Current number of clusters = 3



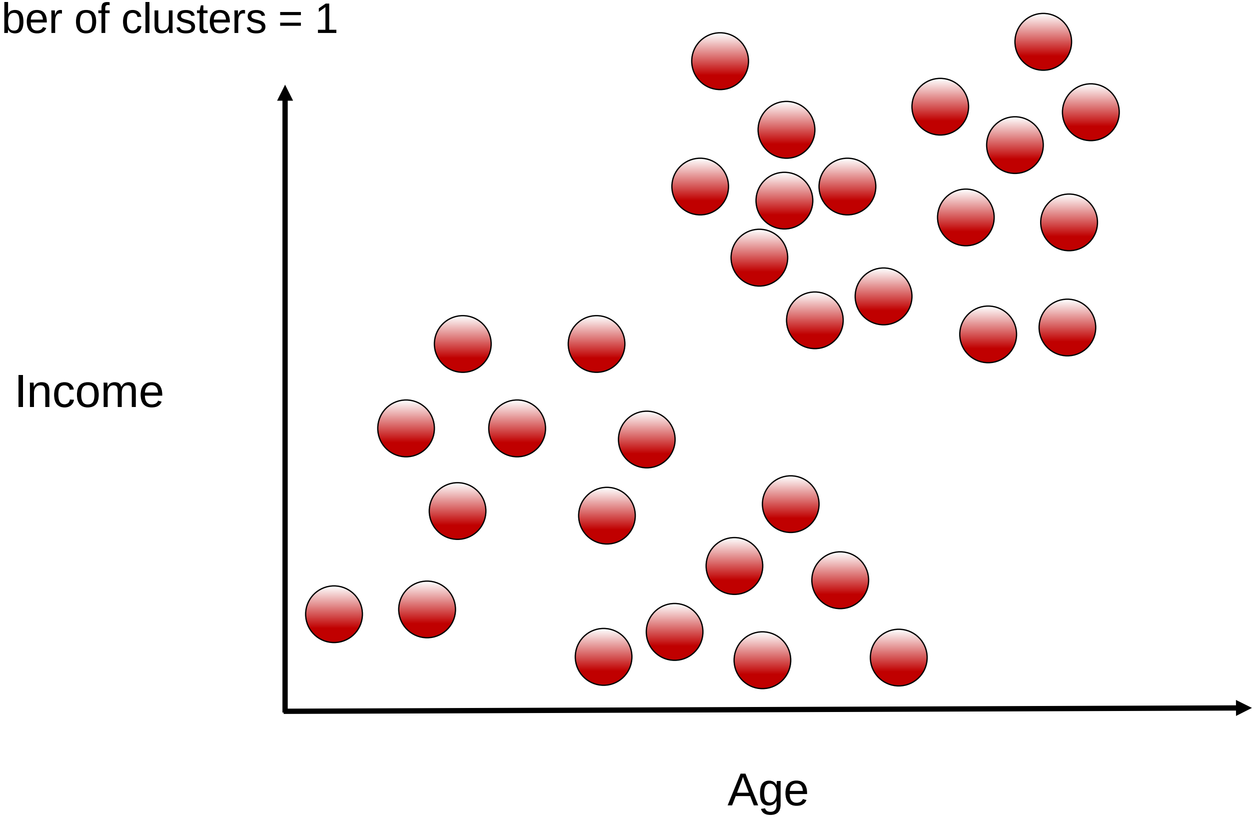
Hierarchical Agglomerative Clustering

Current number of clusters = 2



Hierarchical Agglomerative Clustering

Current number of clusters = 1



Agglomerative Clustering Stopping Conditions

Condition 1	the correct number of clusters is reached

Agglomerative Clustering Stopping Conditions

Condition 1

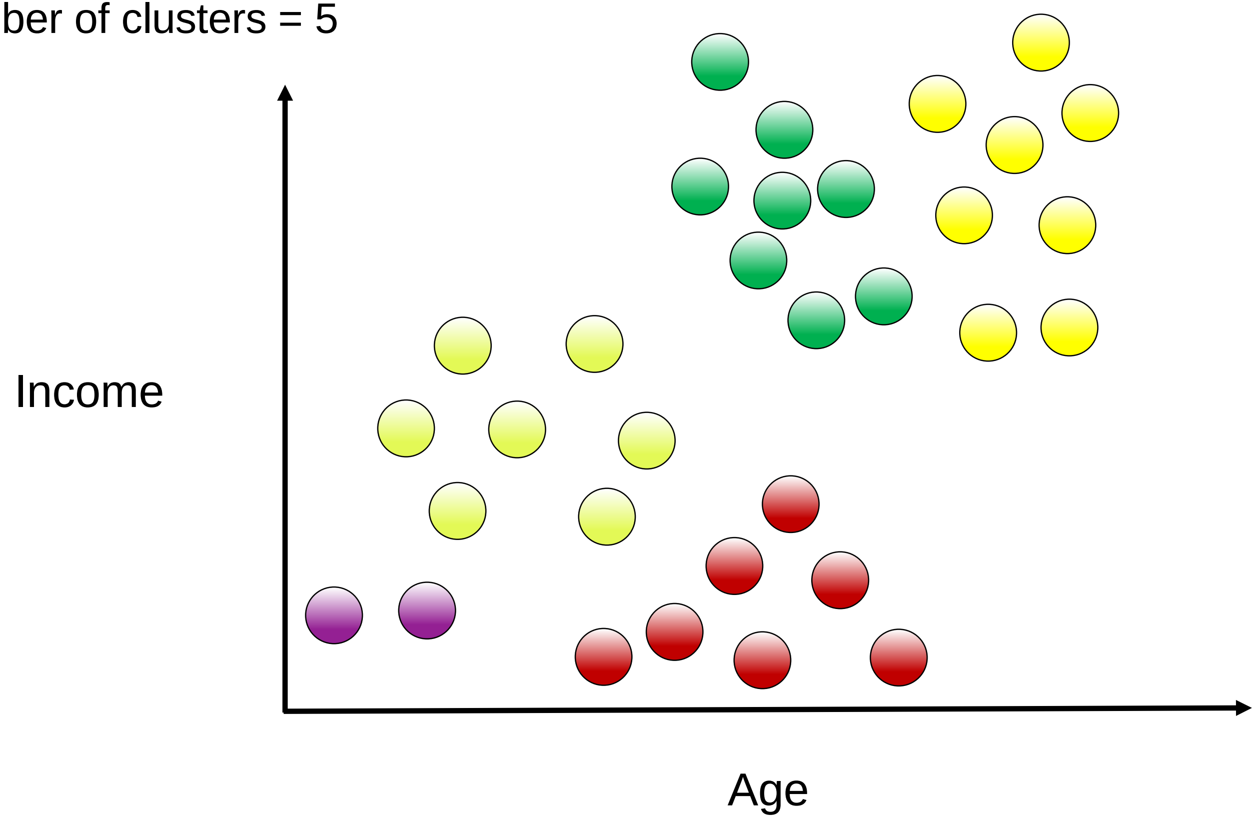
the correct number of clusters is reached

Condition 2

minimum average cluster distance reaches a set value

Hierarchical Agglomerative Clustering

Current number of clusters = 5

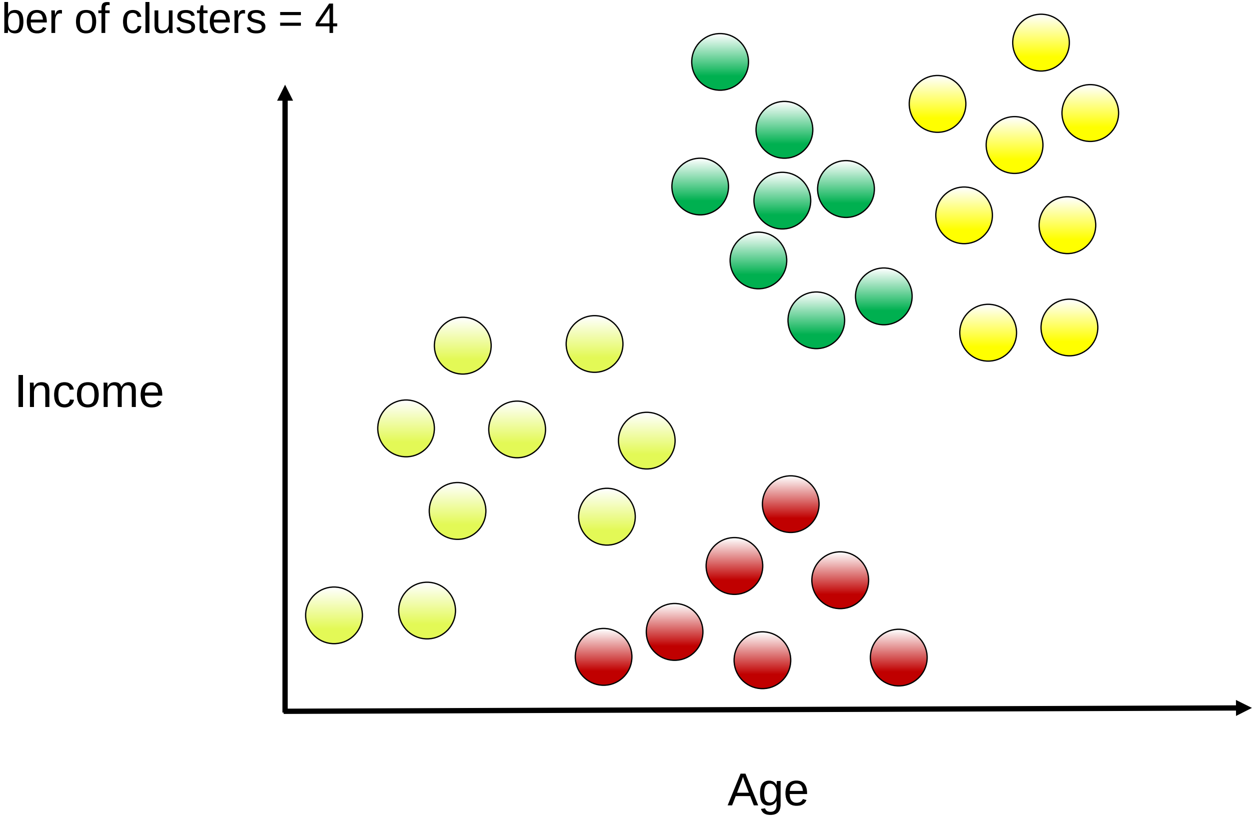


Hierarchical Agglomerative Clustering

Current number of clusters = 5

Hierarchical Agglomerative Clustering

Current number of clusters = 4

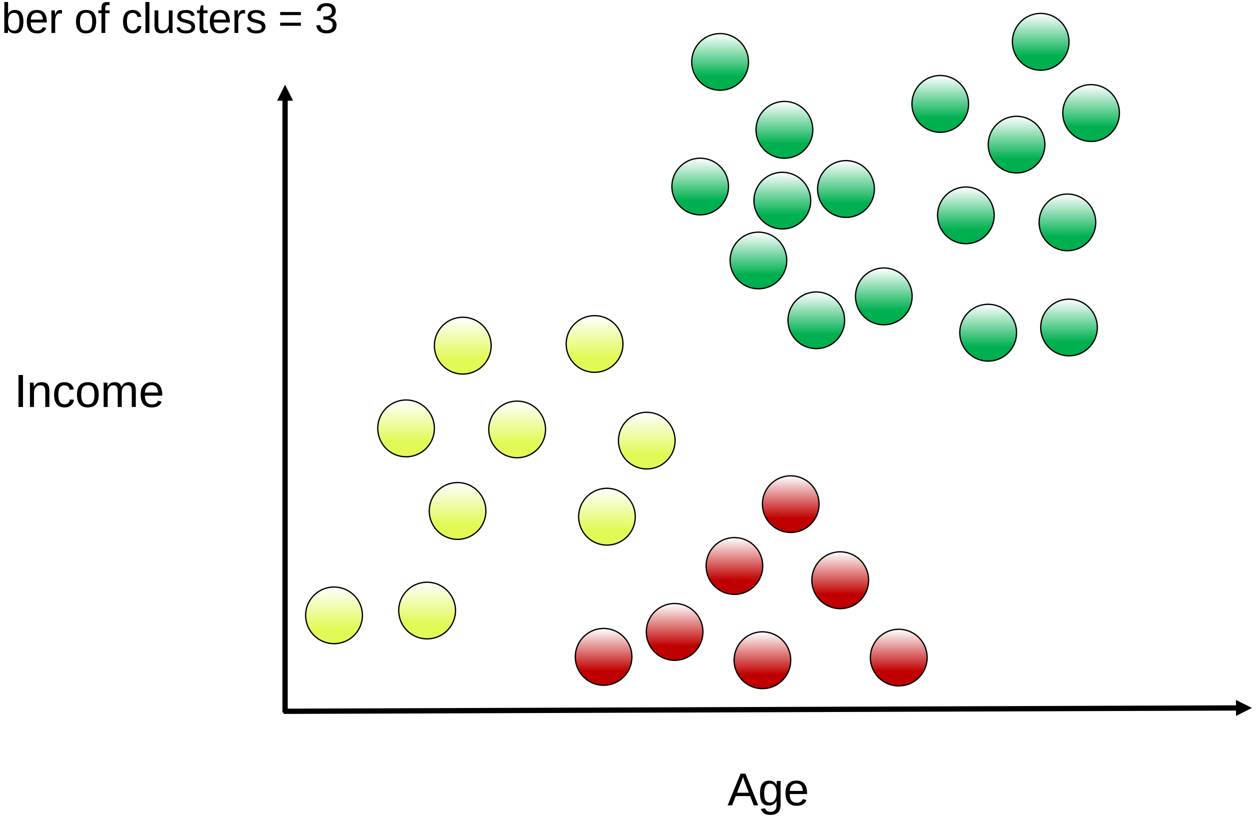


Hierarchical Agglomerative Clustering

Current number of clusters = 4

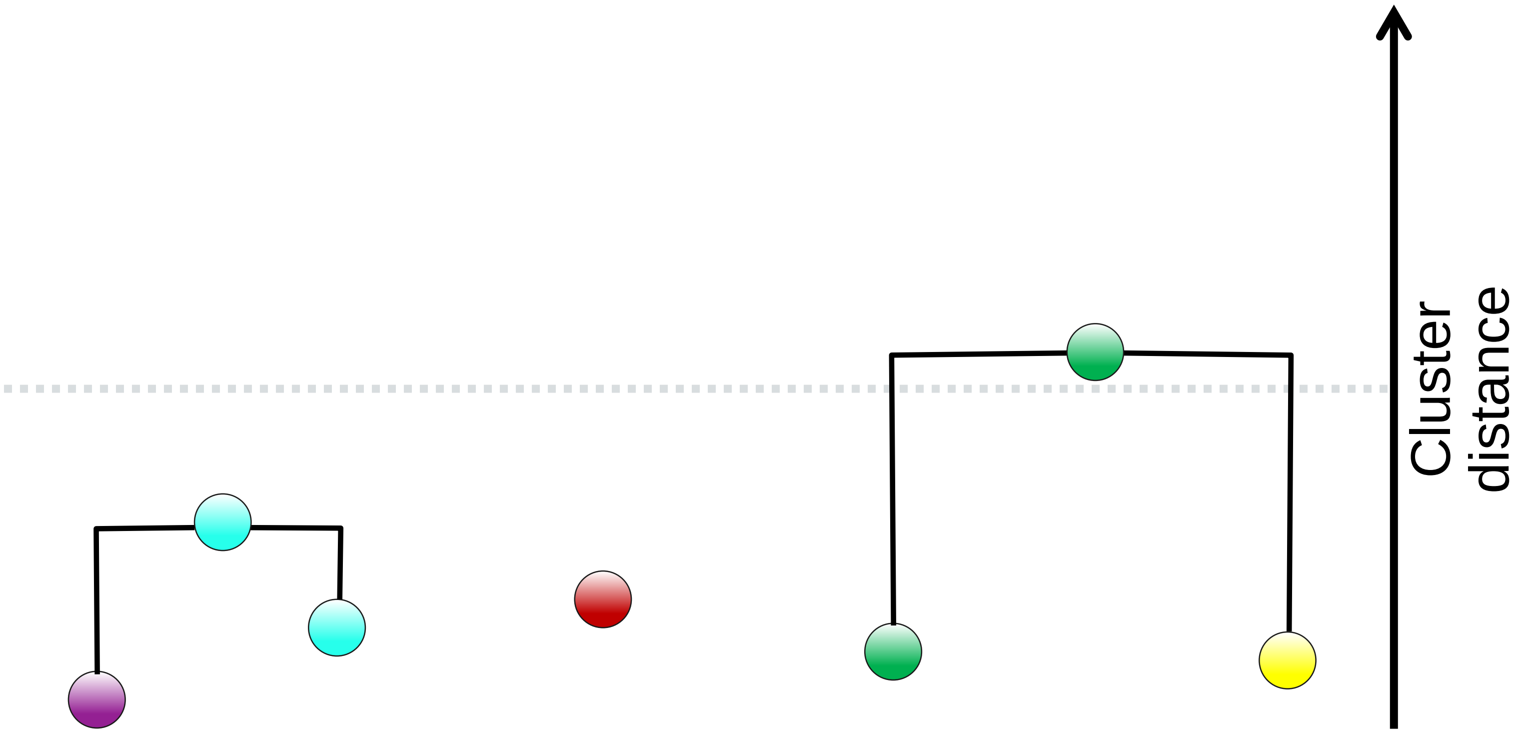
Hierarchical Agglomerative Clustering

Current number of clusters = 3



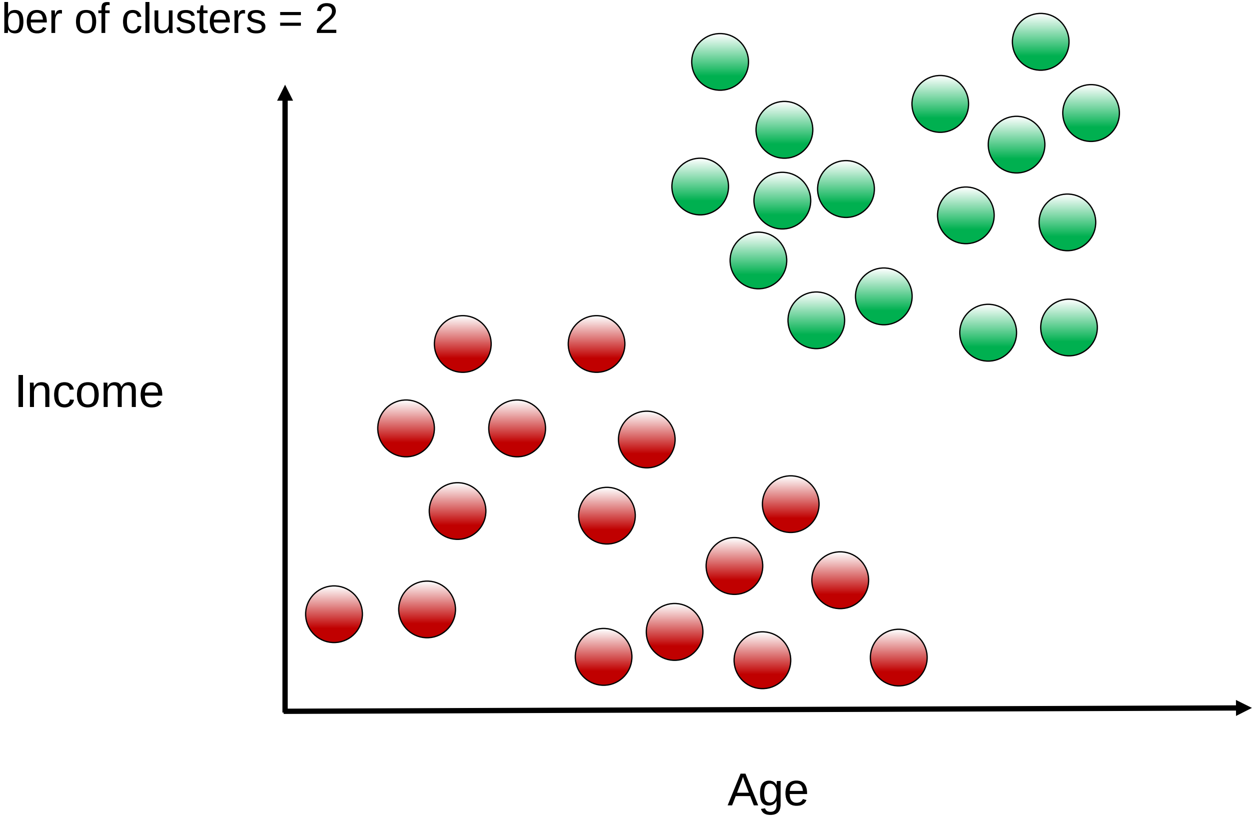
Hierarchical Agglomerative Clustering

Current number of clusters = 3



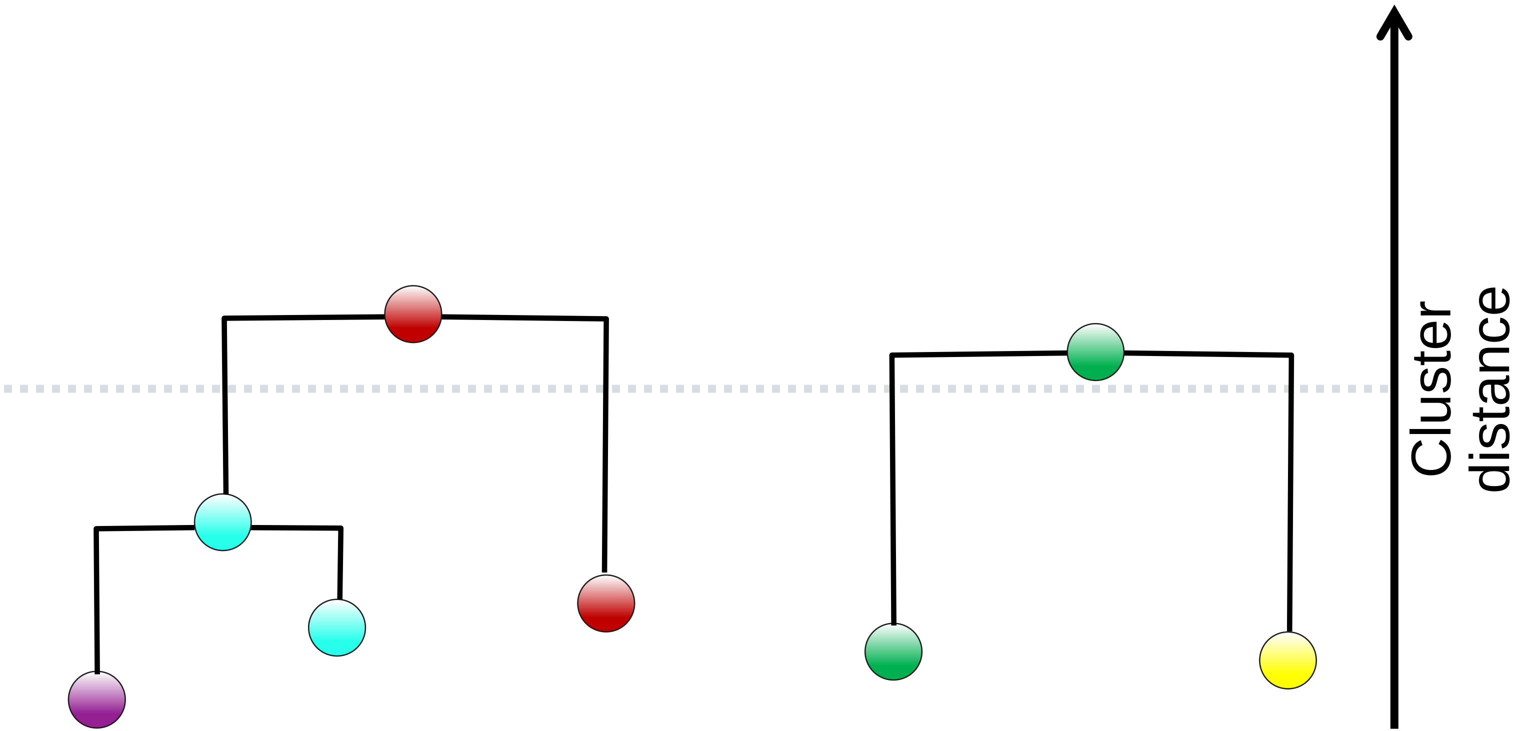
Hierarchical Agglomerative Clustering

Current number of clusters = 2



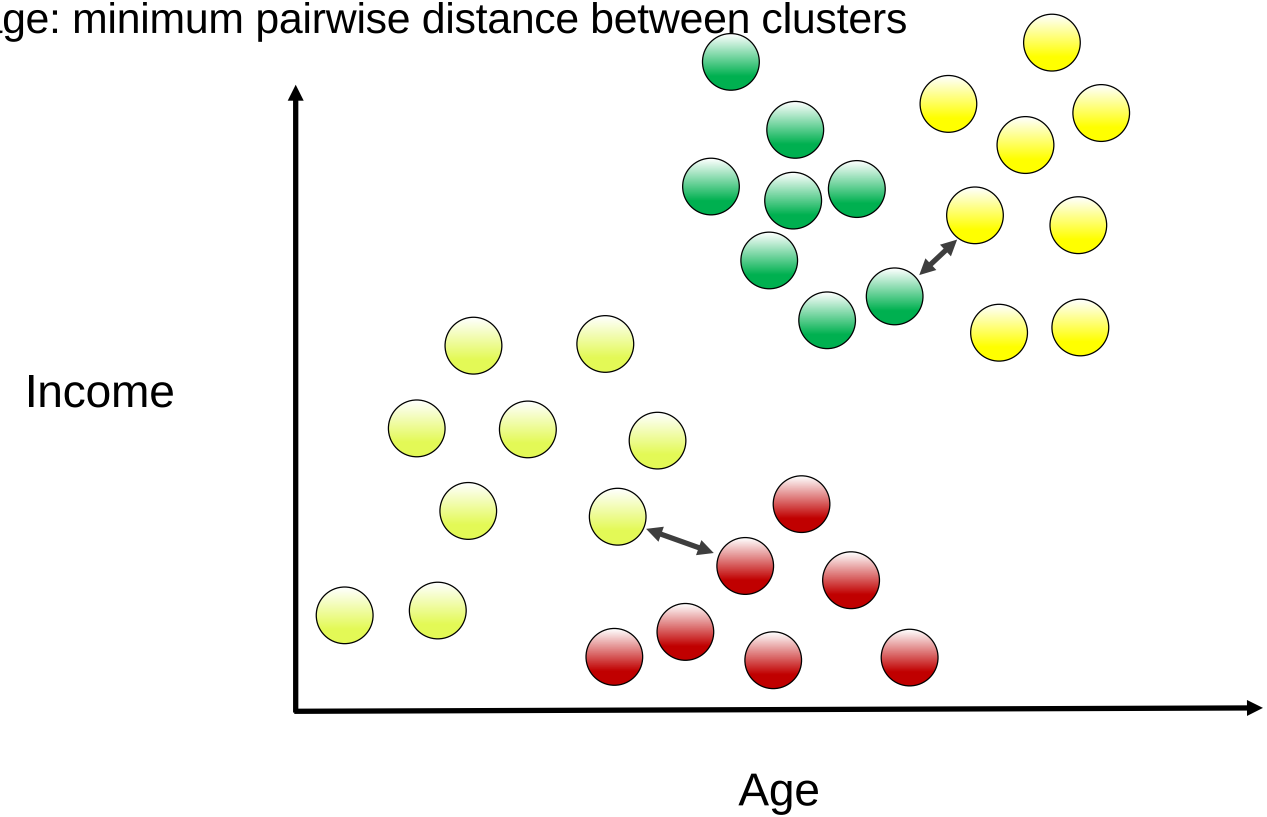
Hierarchical Agglomerative Clustering

Current number of clusters = 2



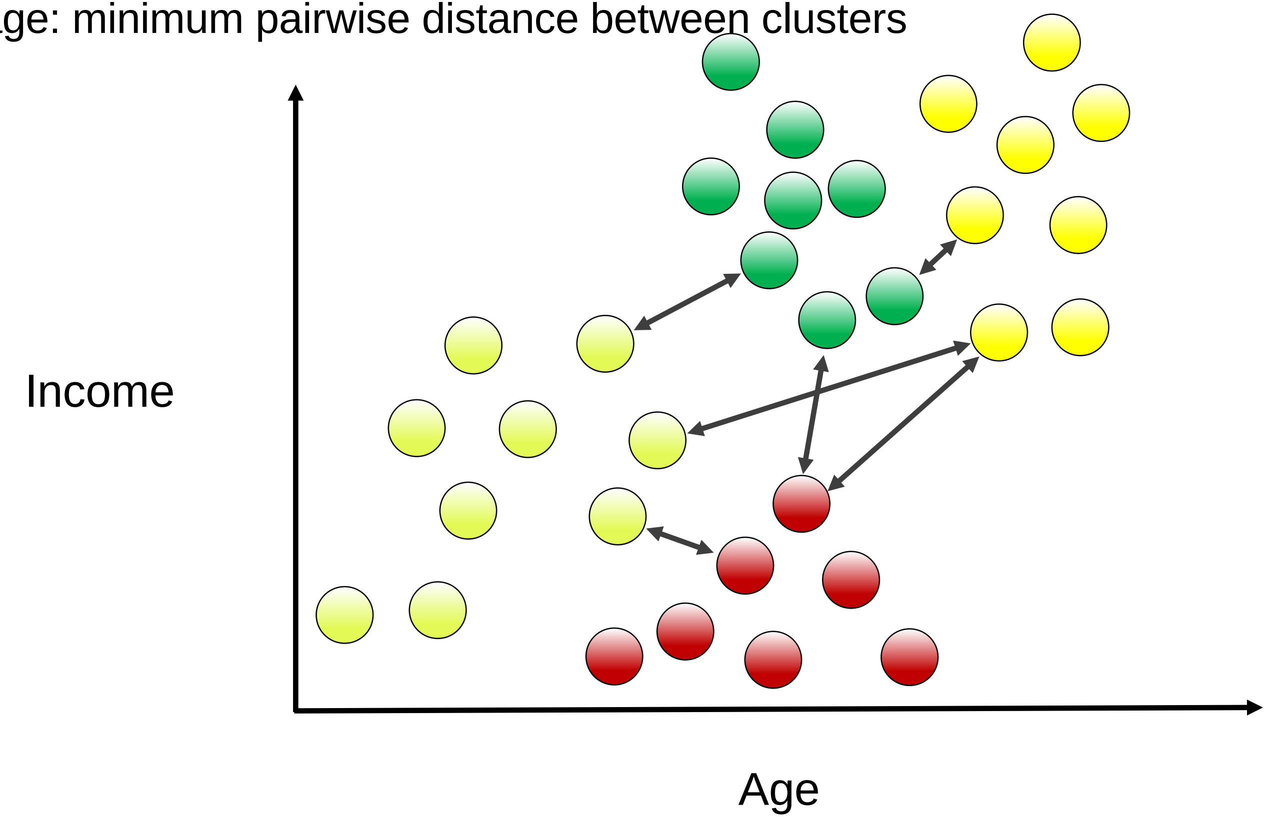
Hierarchical Linkage Types

Single linkage: minimum pairwise distance between clusters



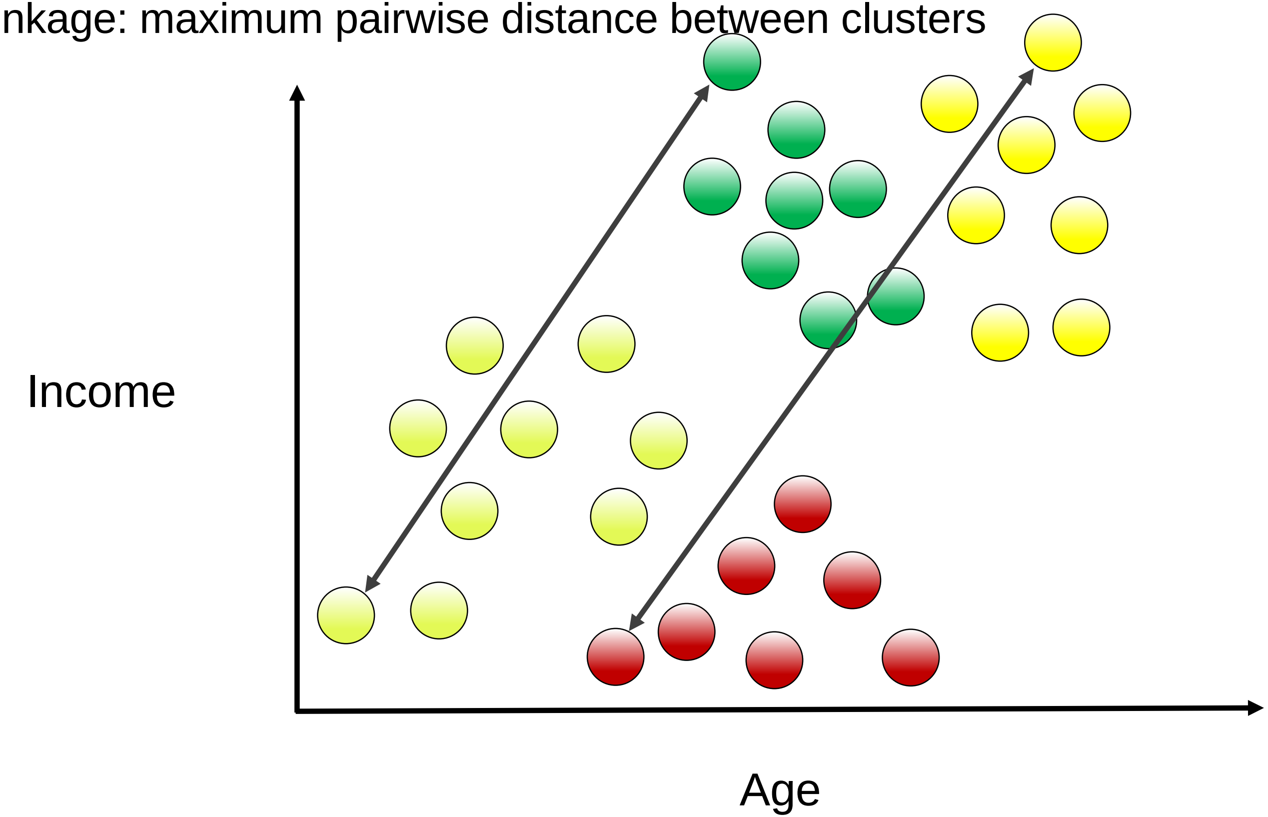
Hierarchical Linkage Types

Single linkage: minimum pairwise distance between clusters



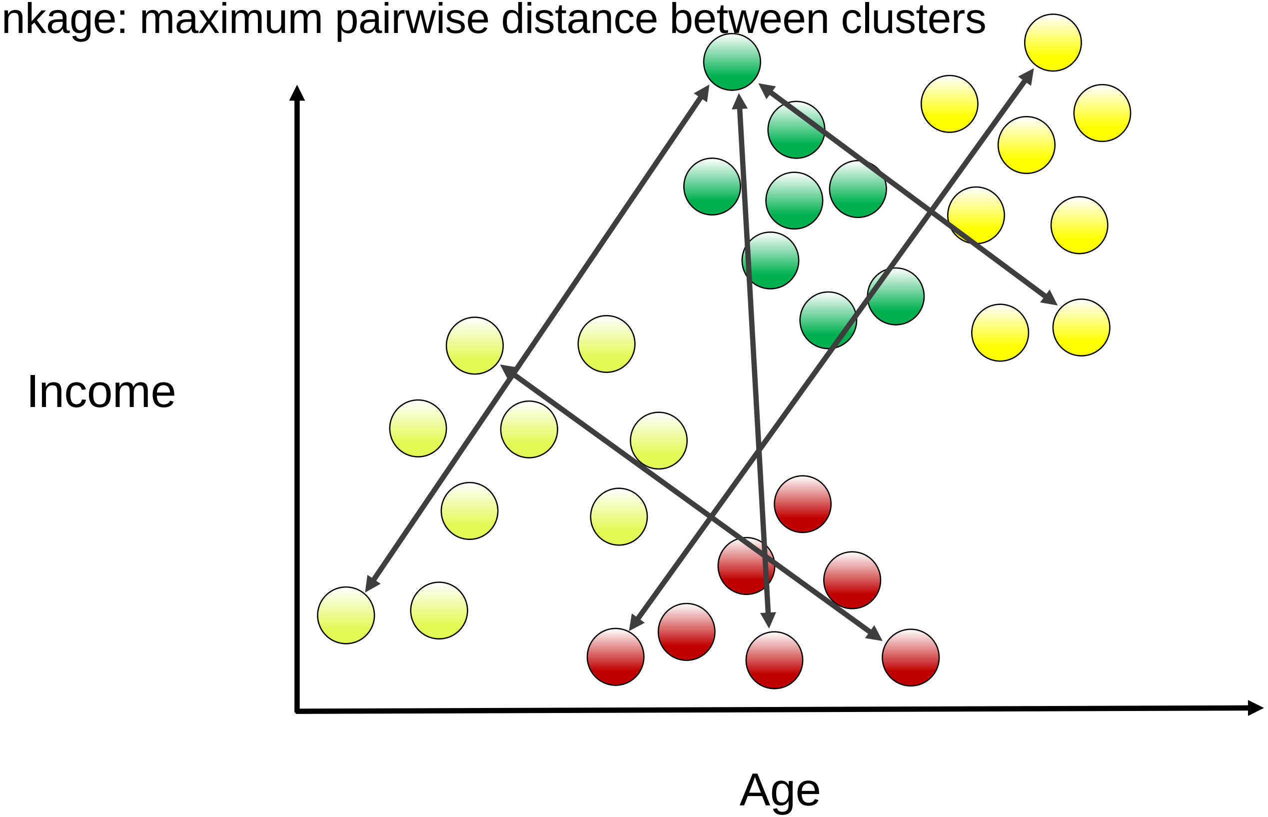
Hierarchical Linkage Types

Complete linkage: maximum pairwise distance between clusters



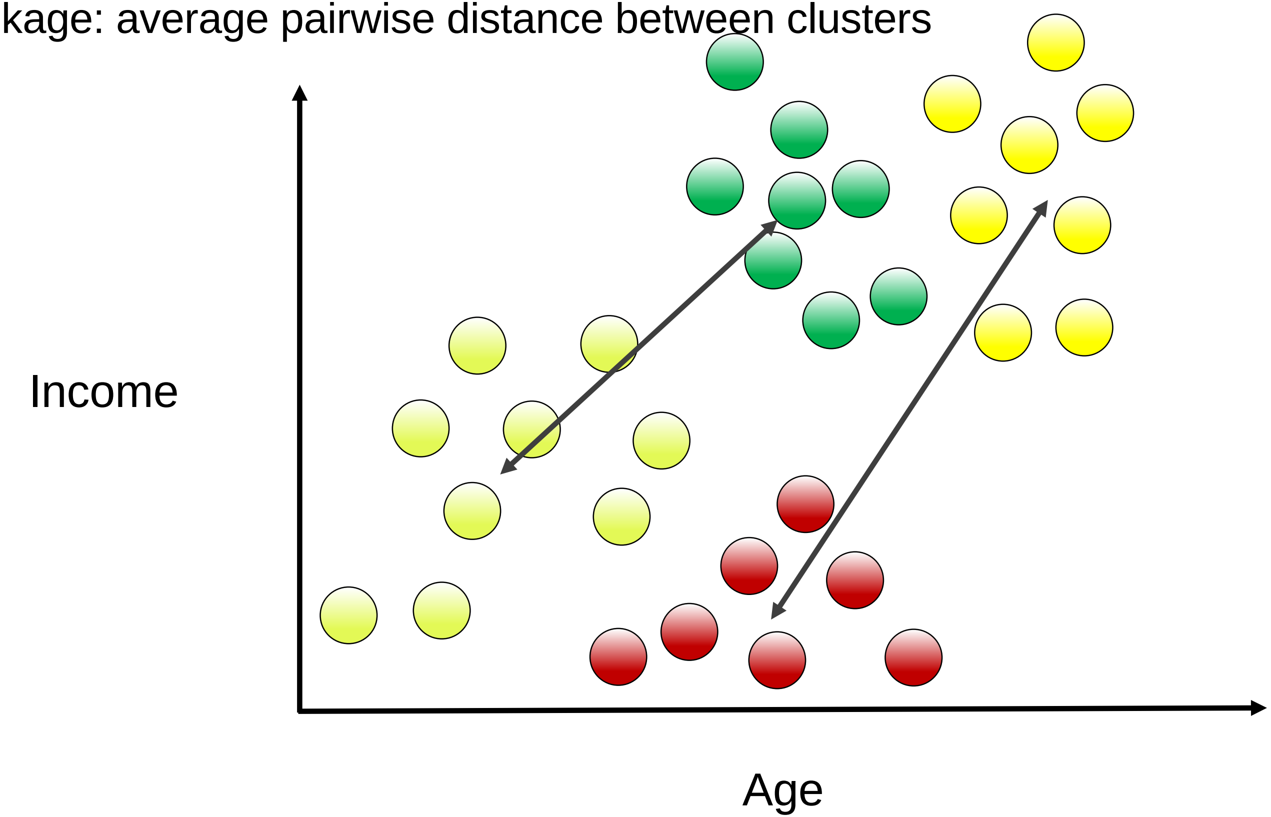
Hierarchical Linkage Types

Complete linkage: maximum pairwise distance between clusters



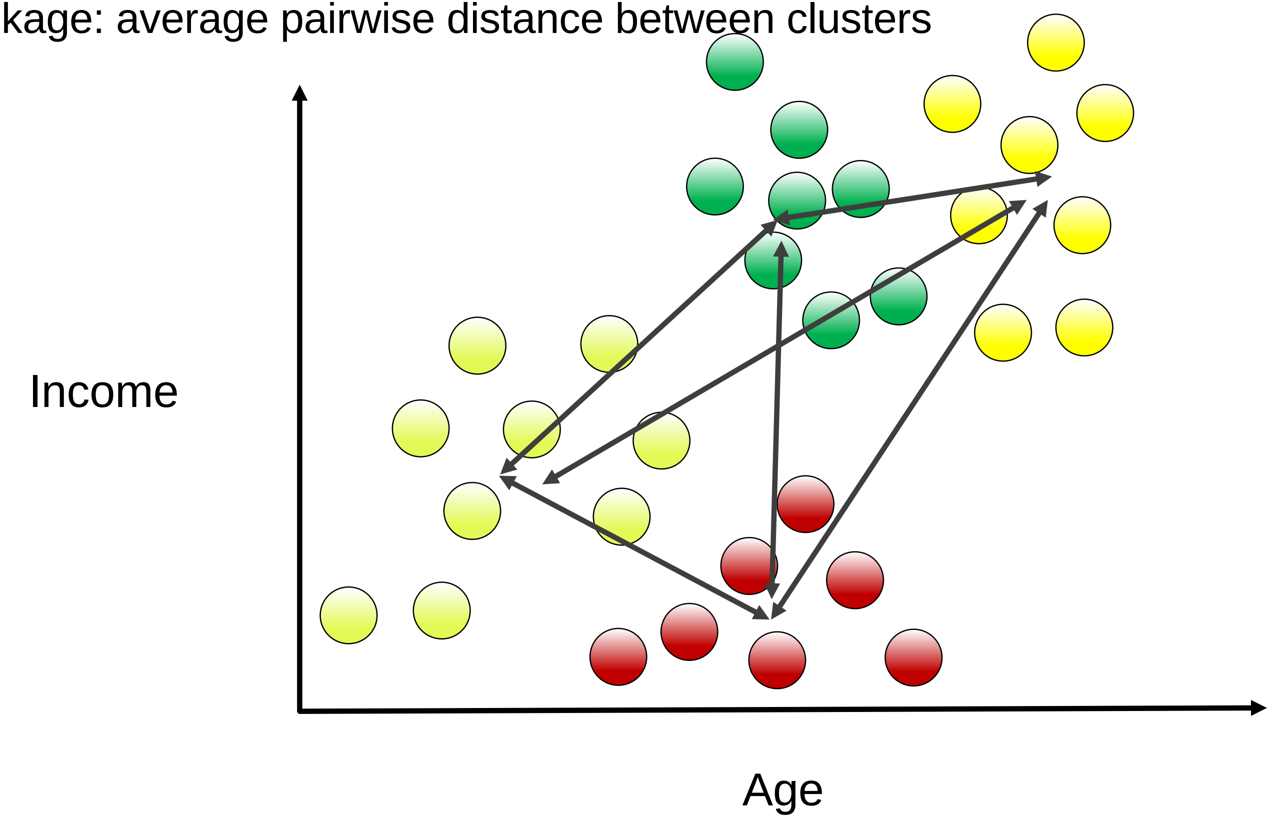
Hierarchical Linkage Types

Average linkage: average pairwise distance between clusters



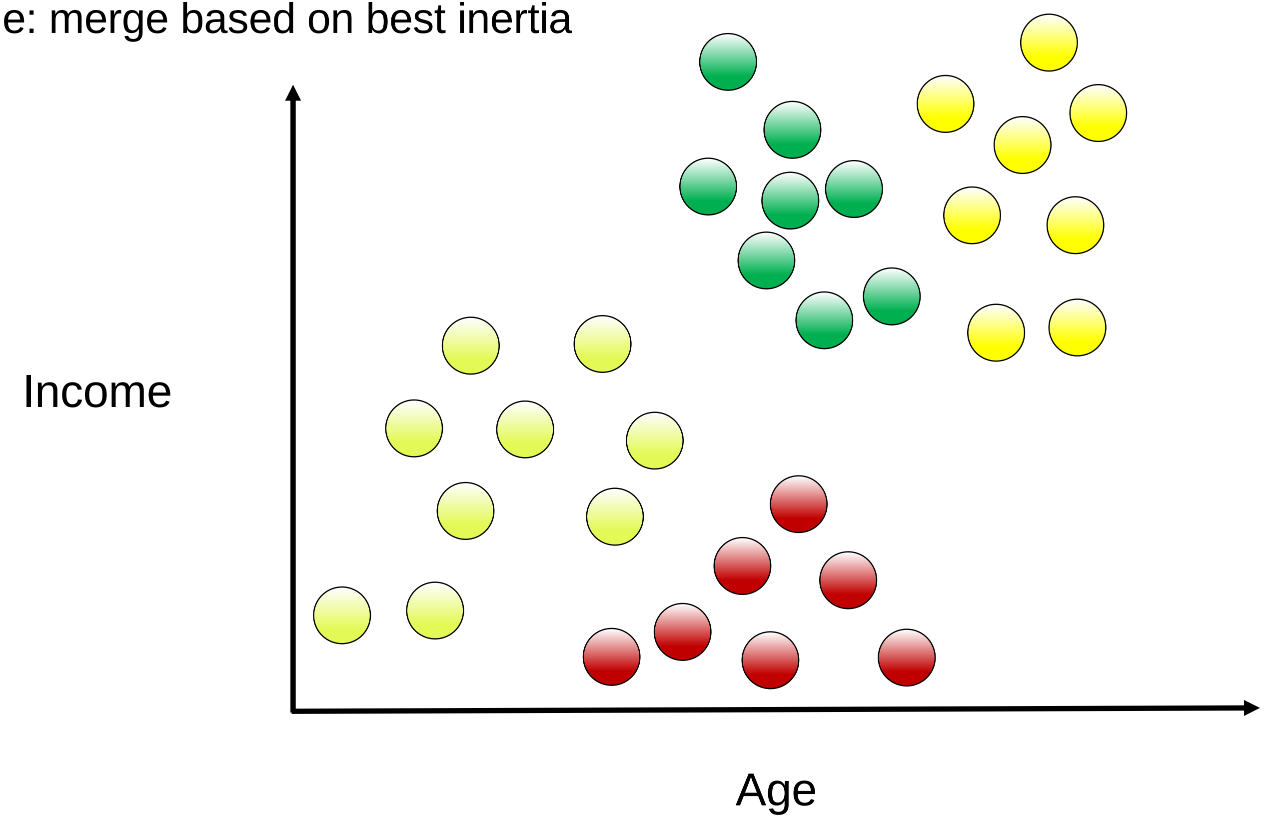
Hierarchical Linkage Types

Average linkage: average pairwise distance between clusters



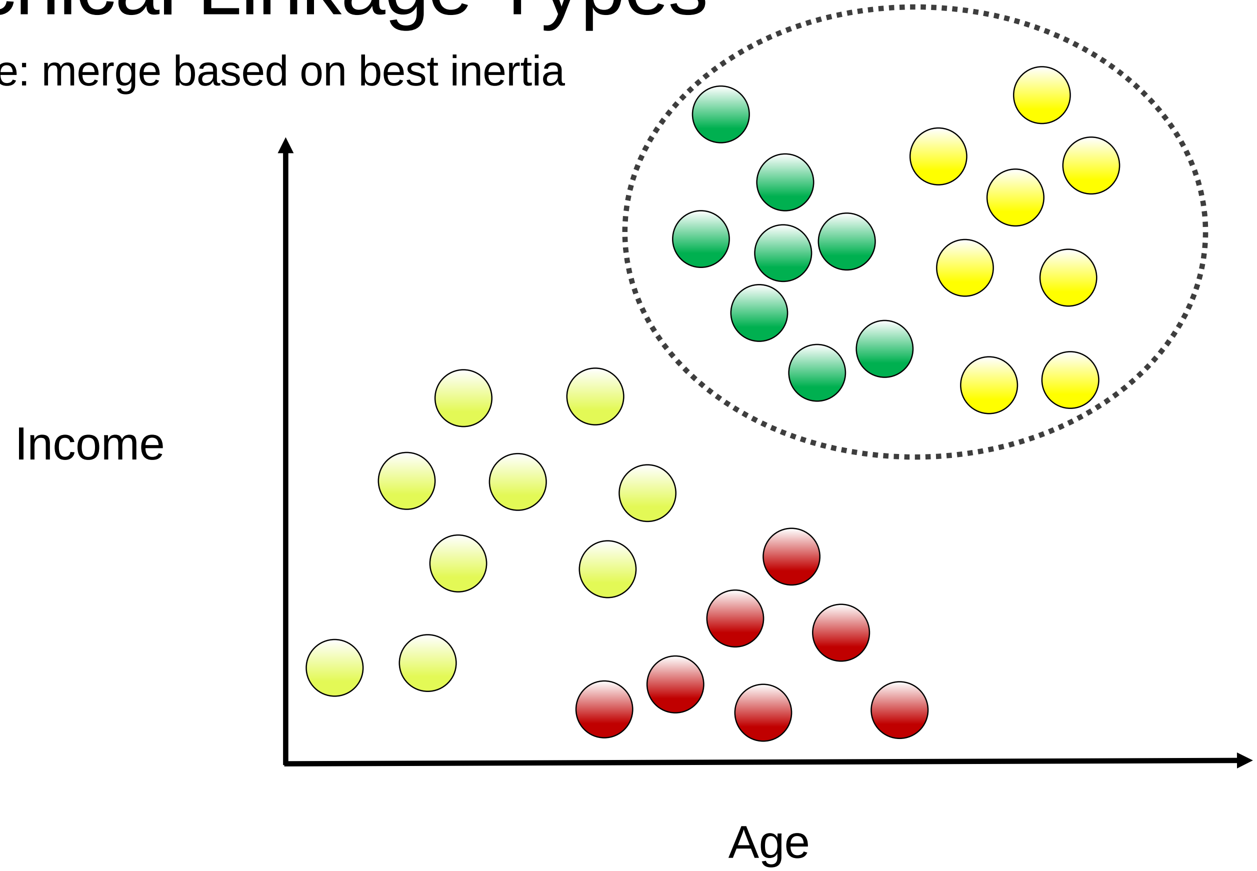
Hierarchical Linkage Types

Ward linkage: merge based on best inertia



Hierarchical Linkage Types

Ward linkage: merge based on best inertia



Agglomerative Clustering: The Syntax

Import the class containing the clustering method

```
from sklearn.cluster import AgglomerativeClustering
```

Create an instance of the class

```
agg = AgglomerativeClustering(n_clusters=3,
                              affinity='euclidean',
                              linkage='ward')
```

Fit the instance on the data and then predict clusters for new data

```
agg = agg.fit(X1)
y_predict = agg.predict(X2)
```

Agglomerative Clustering: The Syntax

Import the class containing the clustering method

```
from sklearn.cluster import AgglomerativeClustering
```

Create an instance of the class

```
agg = AgglomerativeClustering(n_clusters=3,
                              affinity='euclidean',
                              linkage='ward')
```

final number of clusters

Fit the instance on the data and then predict clusters for new data

```
agg = agg.fit(X1)
y_predict = agg.predict(X2)
```

Agglomerative Clustering: The Syntax

Import the class containing the clustering method

```
from sklearn.cluster import AgglomerativeClustering
```

Create an instance of the class

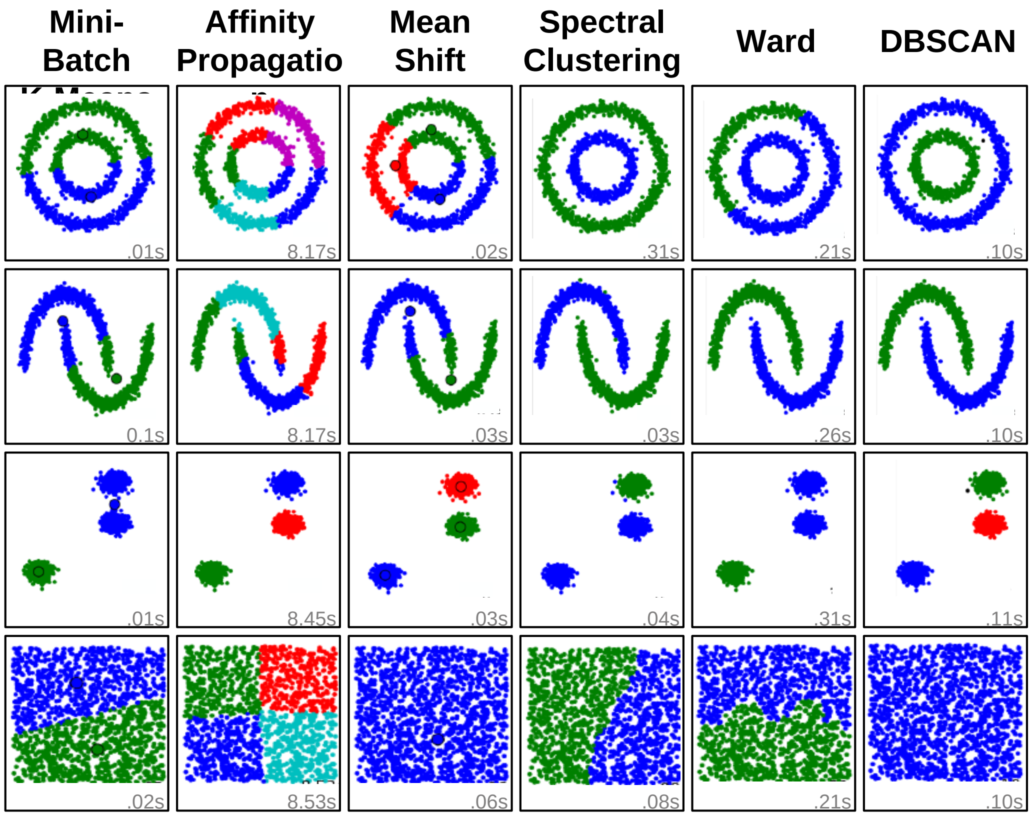
```
agg = AgglomerativeClustering(n_clusters=3,
                              affinity='euclidean',
                              linkage='ward')
```

cluster affinity and aggregation

Fit the instance on the data and then predict clusters for new data

```
agg = agg.fit(X1)
y_predict = agg.predict(X2)
```

Other Types of Clustering



Reference: http://scikit-learn.org/stable/auto_examples/cluster/plot_cluster_comparison.html

Legal Notices and Disclaimers