Clustering

Clustering

- Type of unsupervised learning – label information is not used
- Clustering – partition data into similar groups
- Types of clustering algorithms
  - Partitional. Ex. k-means
  - Hierarchical. Ex. Tree-based clustering

K-means Algorithm

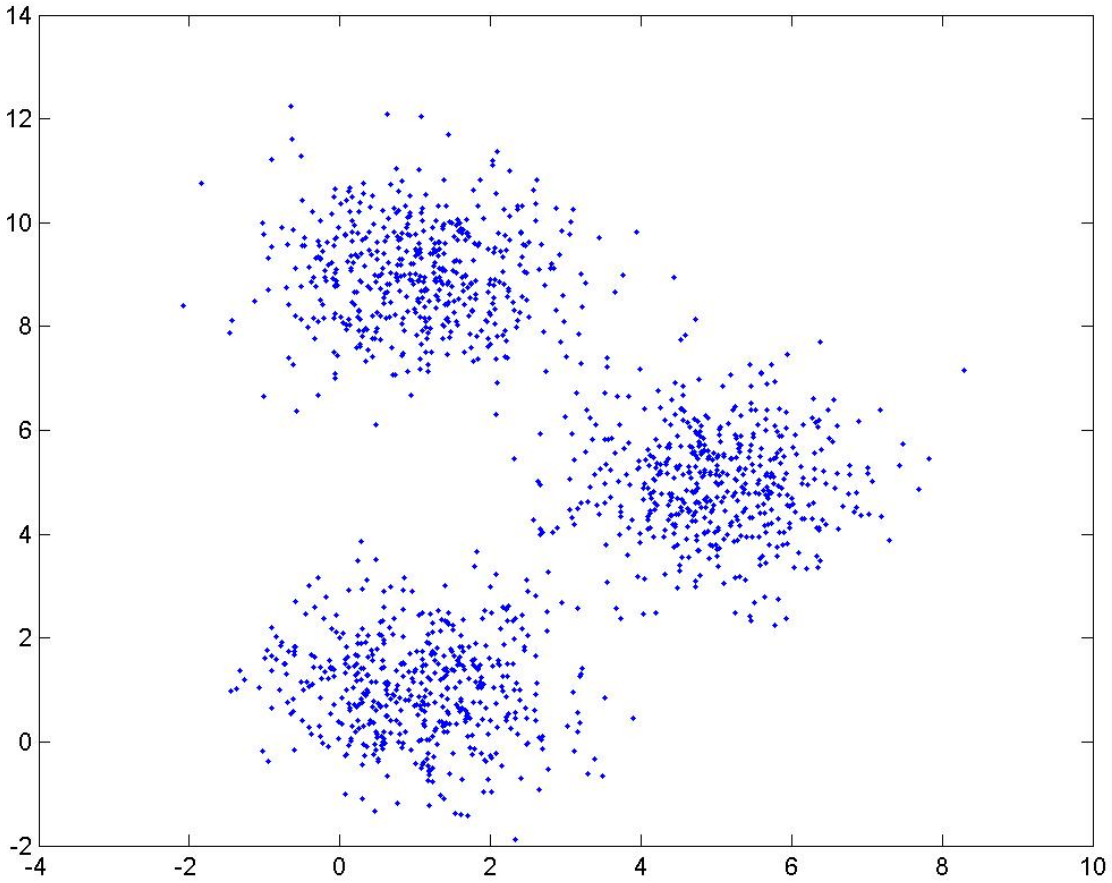
- K-means Algorithm
  - Randomly generate k cluster means m1,...,mk
  - Repeat until convergence
    - Partition data into clusters
      - For each data point x, find the cluster mean that is closest to x (that is, find argmin |x-mi|), assign x to cluster ci
    - Recompute cluster means
      - For i=1 to k, let mi be the mean of cluster ci

K-means Algorithm

- Example. Consider a two-dimensional data set to be partitioned into 3 clusters (that is, k=3)

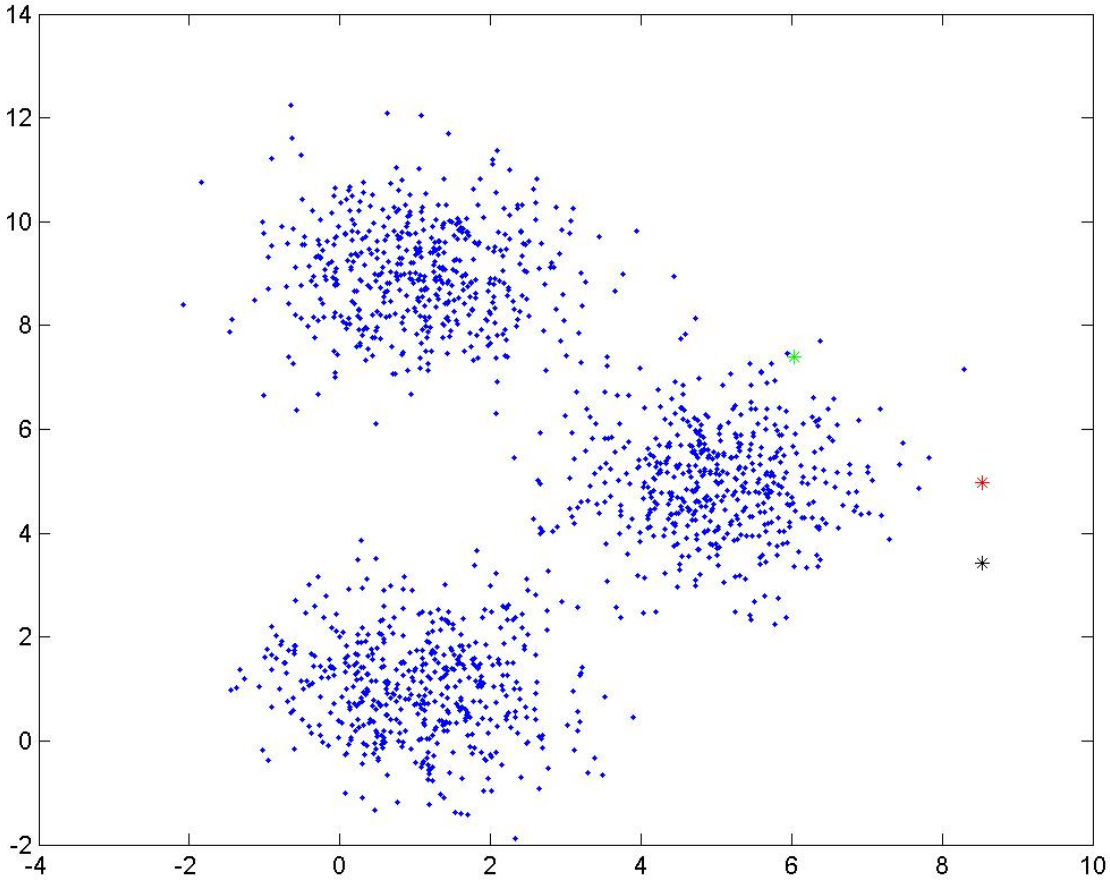
K-means Algorithm

- Original Data



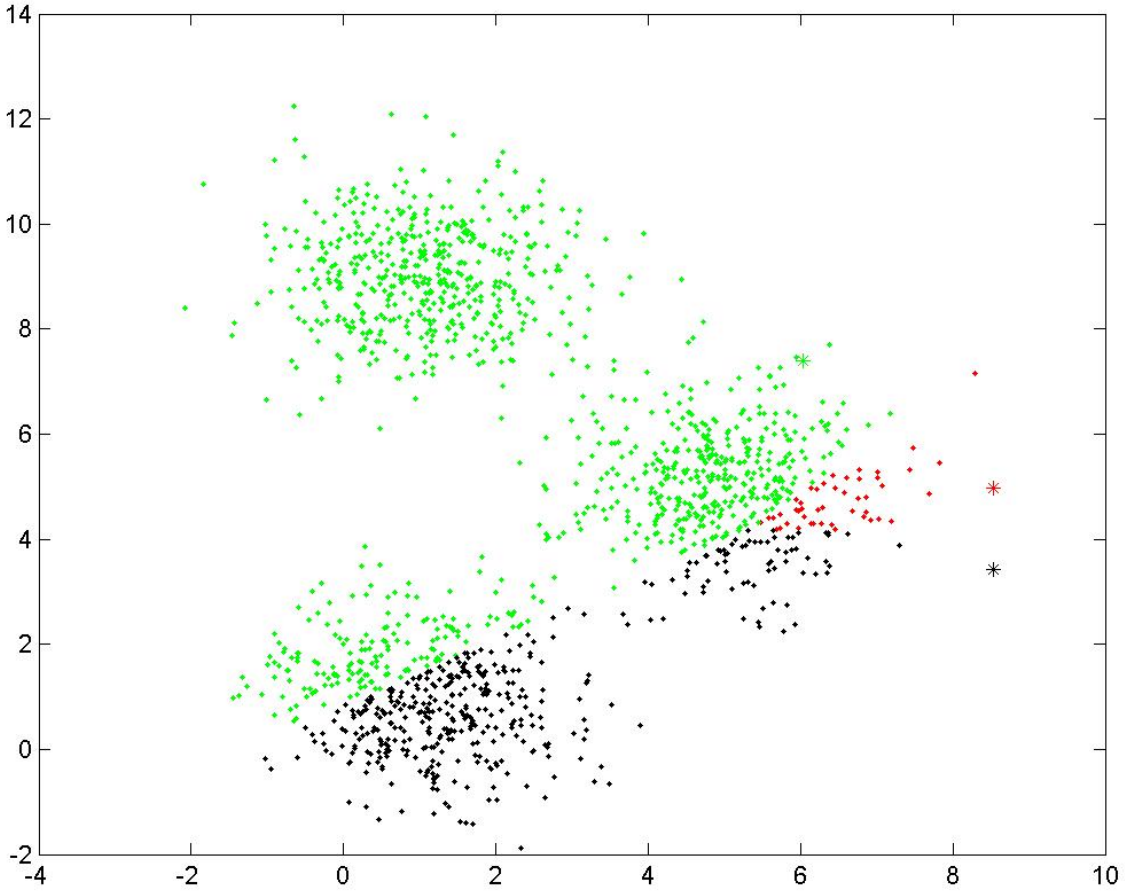
K-means Algorithm

- Original (random) Means



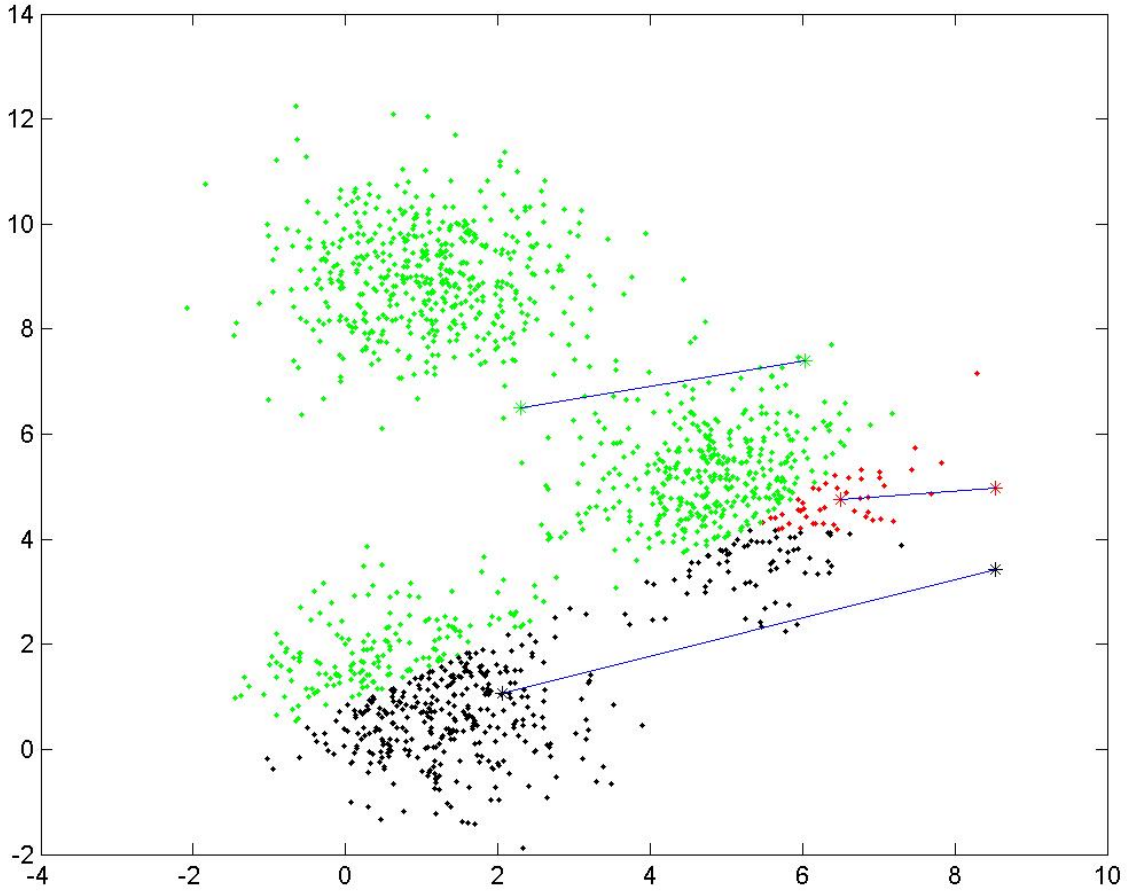
K-means Algorithm

- Iteration 1: Assigning points to clusters



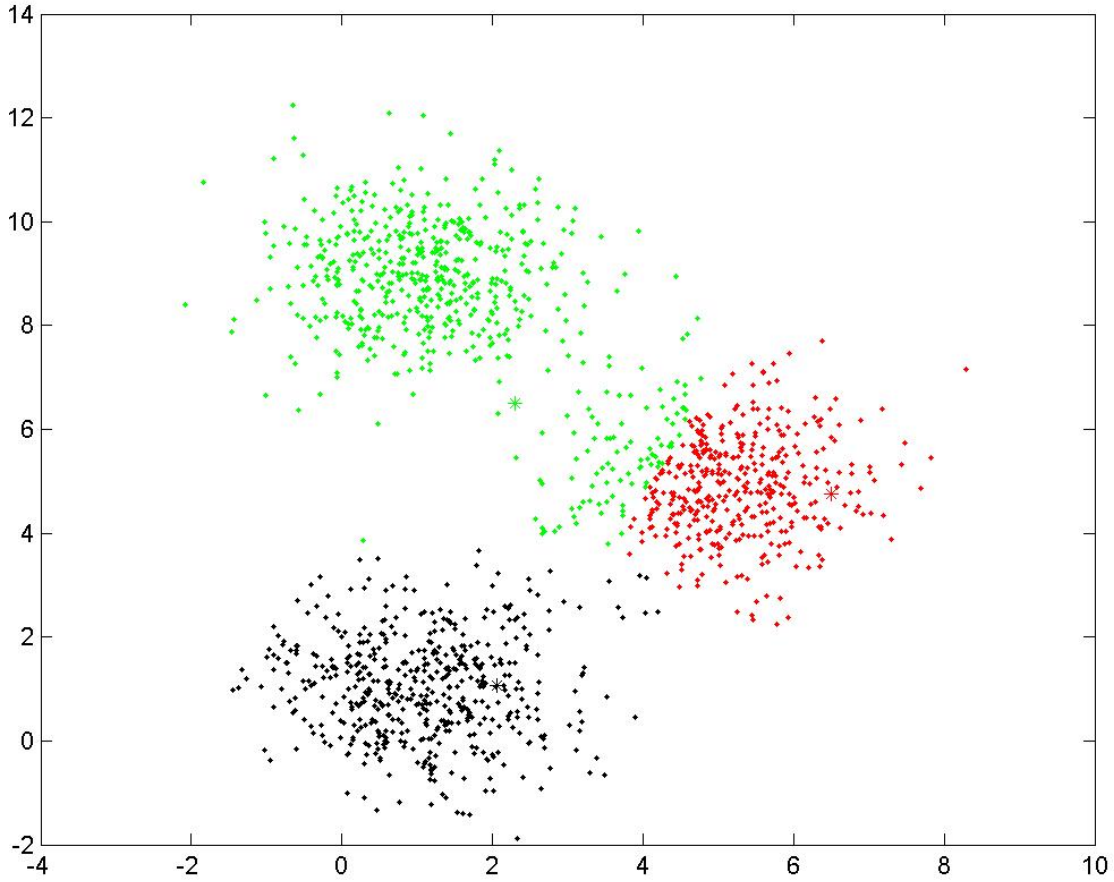
K-means Algorithm

- Iteration 1: Recomputing means



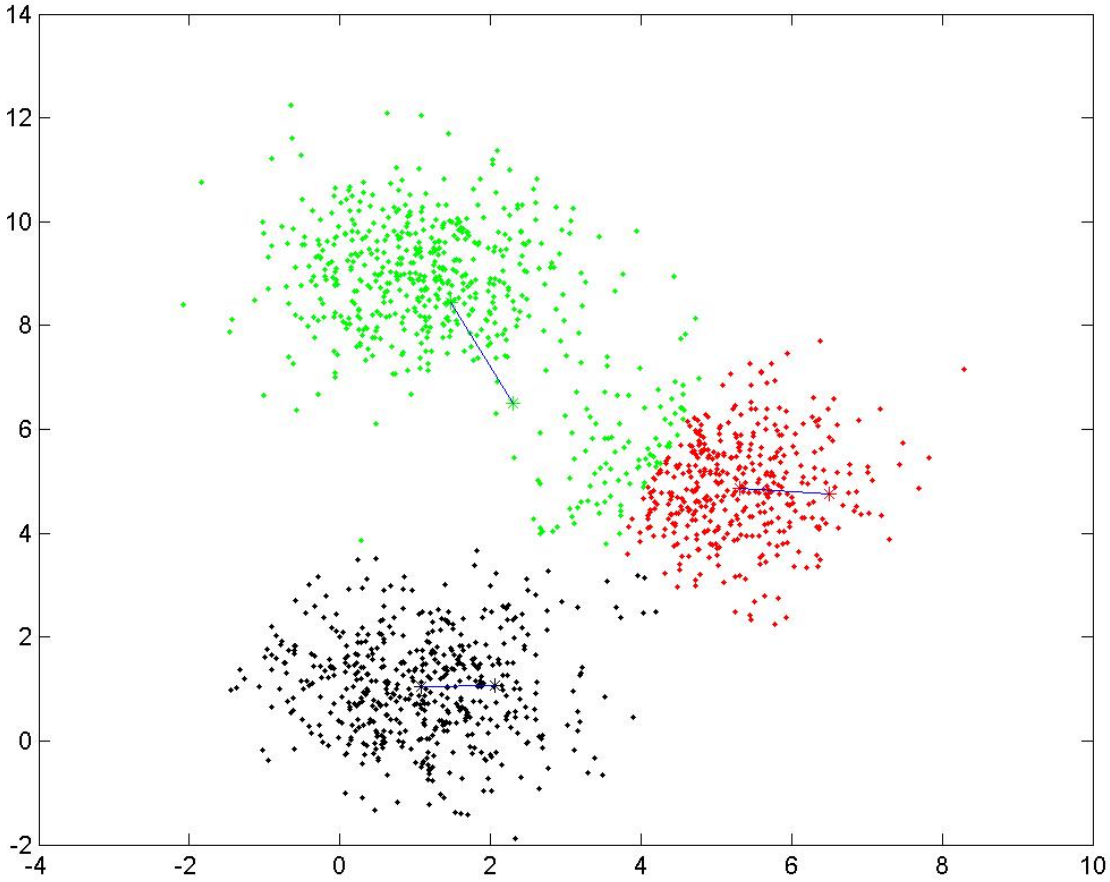
K-means Algorithm

- Iteration 2: Assigning points to clusters



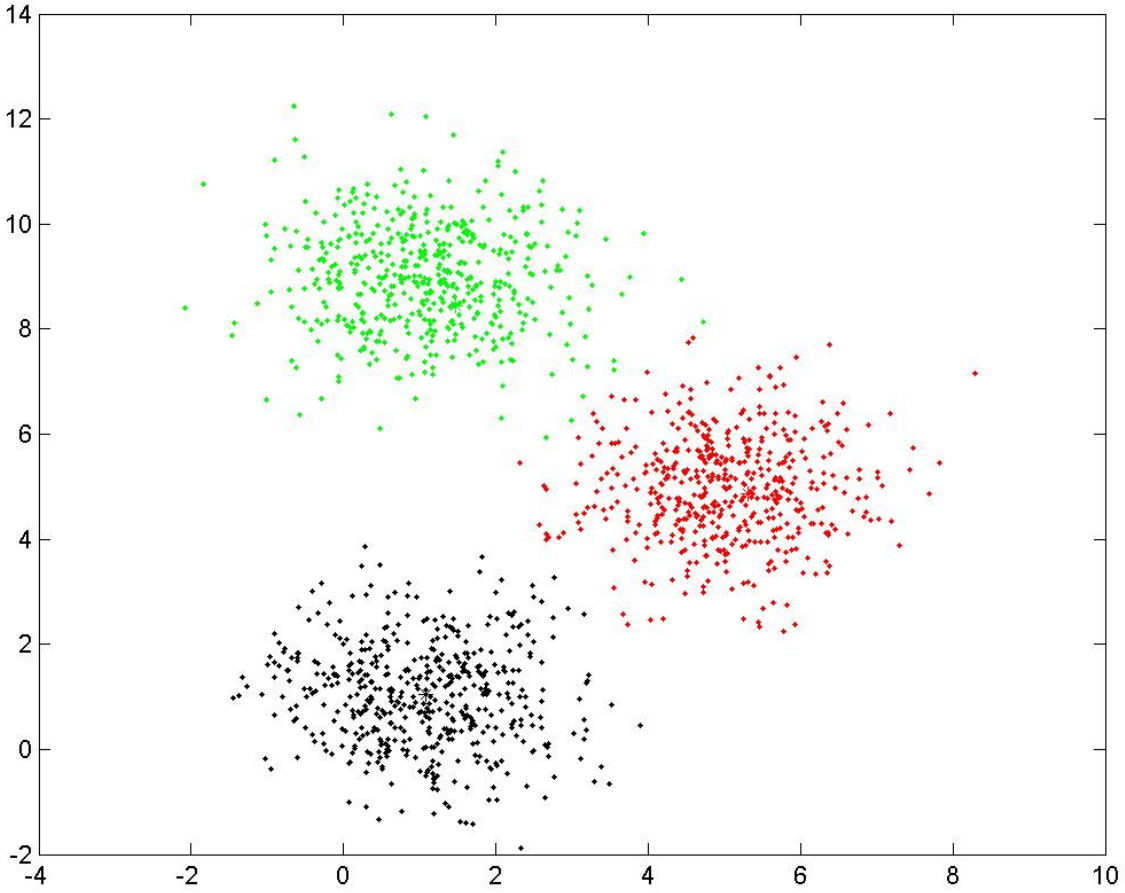
K-means Algorithm

- Iteration 2: Recomputing means



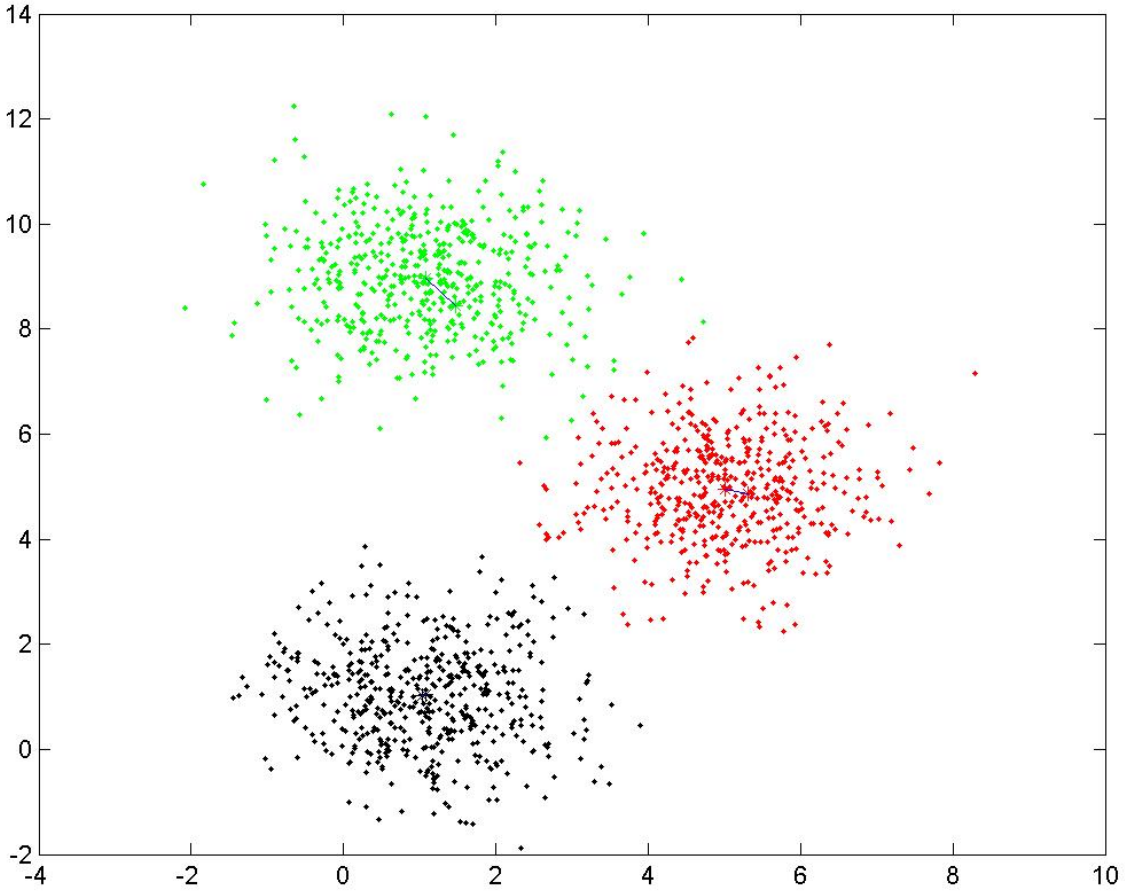
K-means Algorithm

- Iteration 3: Assigning points to clusters



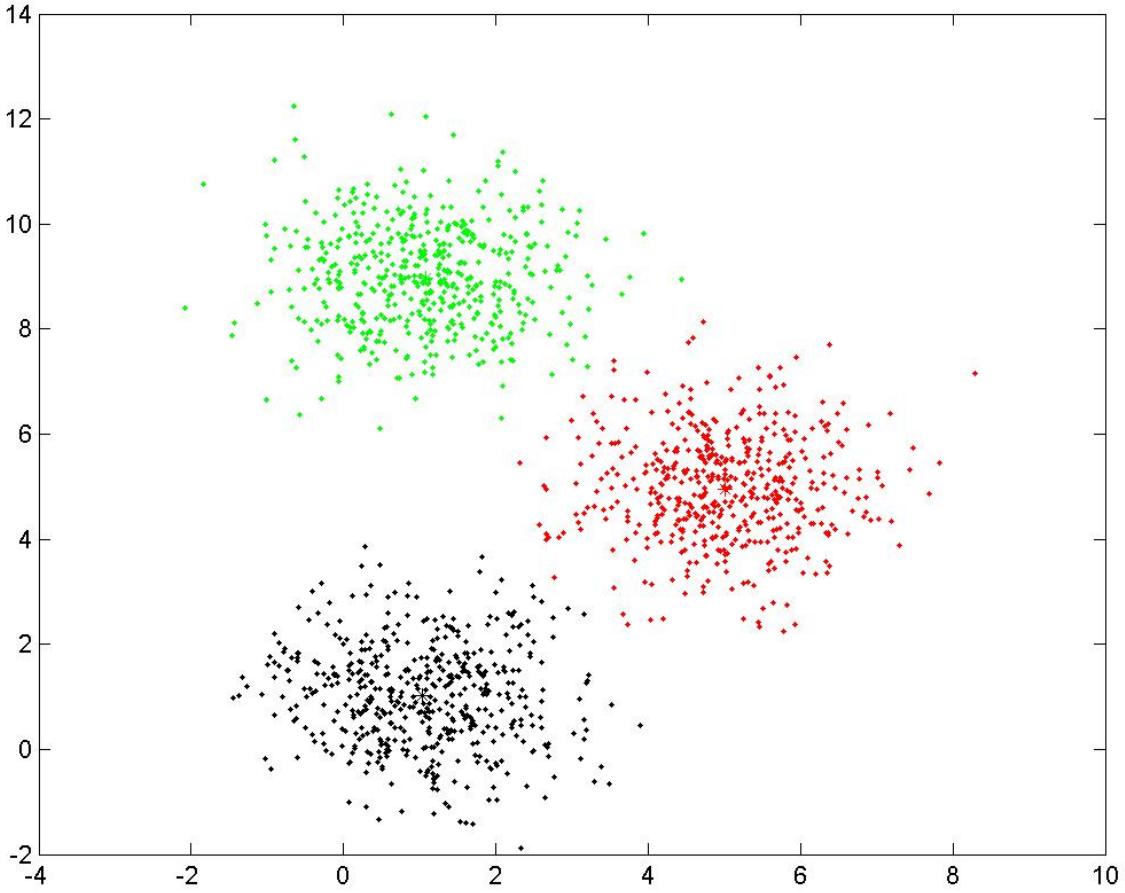
K-means Algorithm

- Iteration 3: Recomputing means



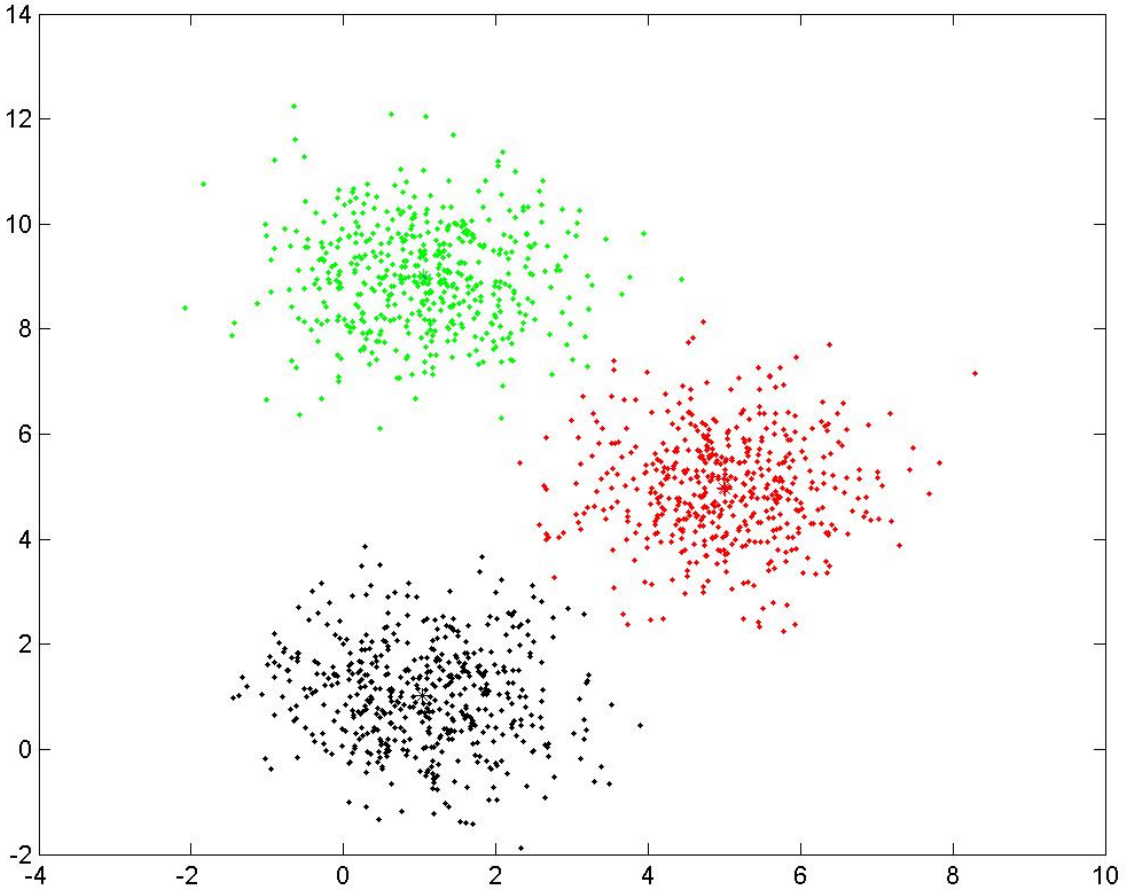
K-means Algorithm

- Iteration 4: Assigning points to clusters



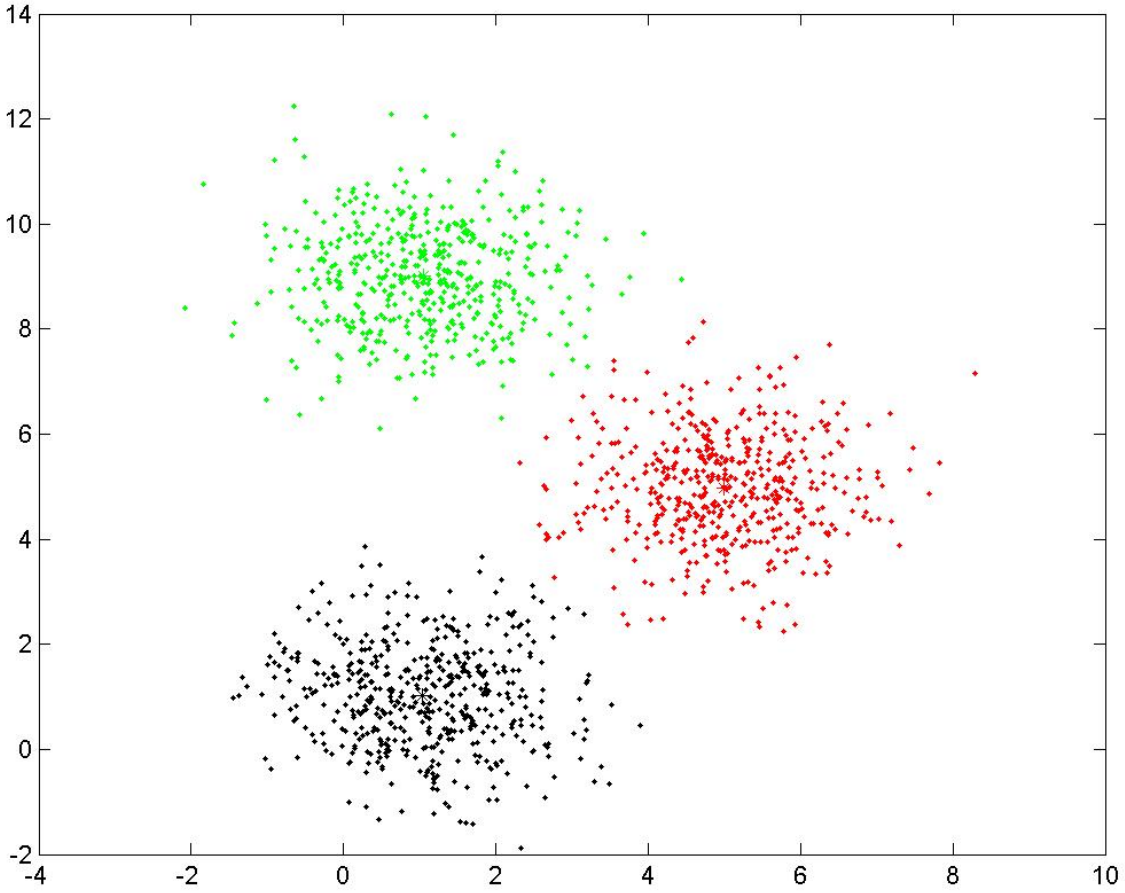
K-means Algorithm

- Iteration 4: Recomputing means



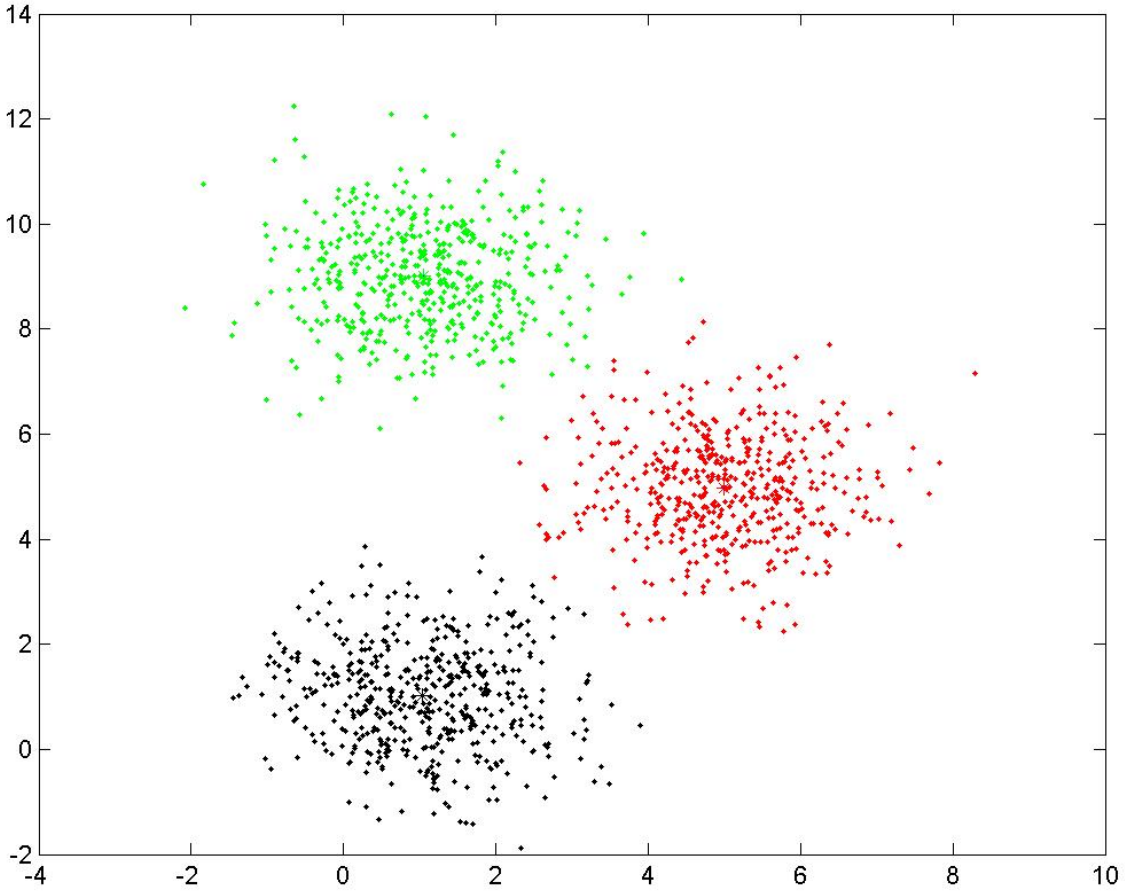
K-means Algorithm

- Iteration 5: Assigning points to clusters



K-means Algorithm

- Iteration 5: Recomputing means



K-means Algorithm

k-means as a pre-processing step:

- Partition data into k clusters
- Compute distance from every point to each of the cluster means
- Use these distances as attributes

```
model = KMeans(n_clusters = k)
model.fit(X)
Xt = model.transform(X)
# Xt has shape (X.shape[0],k)
# where Xt[i,j] is the distance from training example X[i] to cluster center model.cluster_centers_[j]

# The following additional transformation if often performed:
Xt = np.mean(Xt,axis=0) – Xt
Xt = Xt/np.max(Xt,axis=0)
Xt = np.maximum(Xt,0)
# Xt[i,j] is close to 1 if example X[i] is very similar to cluster mean  center
# model.cluster_centers_[j], it will be zero if Xt[i,j] is not  closer to model.cluster_centers_[j] than
# the average example in X
```