

# From Rocchio to K-Means

CS5154/6054

Yizong Cheng

11/1/2022

## 13.1 The text classification problem

DOCUMENT SPACE  
CLASS

TRAINING SET

In text classification, we are given a description  $d \in \mathbb{X}$  of a document, where  $\mathbb{X}$  is the *document space*; and a fixed set of *classes*  $\mathbb{C} = \{c_1, c_2, \dots, c_J\}$ . Classes are also called *categories* or *labels*. Typically, the document space  $\mathbb{X}$  is some type of high-dimensional space, and the classes are human defined for the needs of an application, as in the examples *China* and *documents that talk about multicore computer chips* above. We are given a *training set*  $\mathbb{D}$  of labeled documents  $\langle d, c \rangle$ , where  $\langle d, c \rangle \in \mathbb{X} \times \mathbb{C}$ . For example:

$$\langle d, c \rangle = \langle \text{Beijing joins the World Trade Organization}, \text{China} \rangle$$

for the one-sentence document *Beijing joins the World Trade Organization* and the class (or label) *China*.

LEARNING METHOD  
CLASSIFIER

Using a *learning method* or *learning algorithm*, we then wish to learn a classifier or *classification function*  $\gamma$  that maps documents to classes:

(13.1)

$$\gamma : \mathbb{X} \rightarrow \mathbb{C}$$

```
TRAINROCCHIO( $\mathbb{C}, \mathbb{D}$ )  
1  for each  $c_j \in \mathbb{C}$   
2  do  $D_j \leftarrow \{d : \langle d, c_j \rangle \in \mathbb{D}\}$   
3      $\vec{\mu}_j \leftarrow \frac{1}{|D_j|} \sum_{d \in D_j} \vec{v}(d)$   
4  return  $\{\vec{\mu}_1, \dots, \vec{\mu}_J\}$ 
```

```
APPLYROCCHIO( $\{\vec{\mu}_1, \dots, \vec{\mu}_J\}, d$ )  
1  return  $\arg \min_j |\vec{\mu}_j - \vec{v}(d)|$ 
```

► Figure 14.4 Rocchio classification: Training and testing.

# ApplyRocchio to D, the Training Set?

- Given vectorized documents D from the document space X.
  - And an initial class assignment  $\gamma : X \rightarrow C$ .
- TrainRocchio produces one centroid for each class in C.
  - $\mu_c$  for c in C
- ApplyRocchio updates the class assignment to all documents as  $\operatorname{argmin}_c |\mu_c - d|$ .
- Repeat TrainRocchio and ApplyRocchio on D until no more changes in class assignment.
  - May start with an arbitrary centroid selection and then ApplyRocchio.

```

K-MEANS( $\{\vec{x}_1, \dots, \vec{x}_N\}, K$ )
1   $(\vec{s}_1, \vec{s}_2, \dots, \vec{s}_K) \leftarrow \text{SELECTRANDOMSEEDS}(\{\vec{x}_1, \dots, \vec{x}_N\}, K)$ 
2  for  $k \leftarrow 1$  to  $K$ 
3  do  $\vec{\mu}_k \leftarrow \vec{s}_k$ 
4  while stopping criterion has not been met
5  do for  $k \leftarrow 1$  to  $K$ 
6      do  $\omega_k \leftarrow \{\}$ 
7      for  $n \leftarrow 1$  to  $N$ 
8          do  $j \leftarrow \arg \min_{j'} |\vec{\mu}_{j'} - \vec{x}_n|$ 
9               $\omega_j \leftarrow \omega_j \cup \{\vec{x}_n\}$  (reassignment of vectors)
10     for  $k \leftarrow 1$  to  $K$ 
11         do  $\vec{\mu}_k \leftarrow \frac{1}{|\omega_k|} \sum_{\vec{x} \in \omega_k} \vec{x}$  (recomputation of centroids)
12 return  $\{\vec{\mu}_1, \dots, \vec{\mu}_K\}$ 

```

► **Figure 16.5** The *K-means* algorithm. For most IR applications, the vectors  $\vec{x}_n \in \mathbb{R}^M$  should be length-normalized. Alternative methods of seed selection and initialization are discussed on page 364.

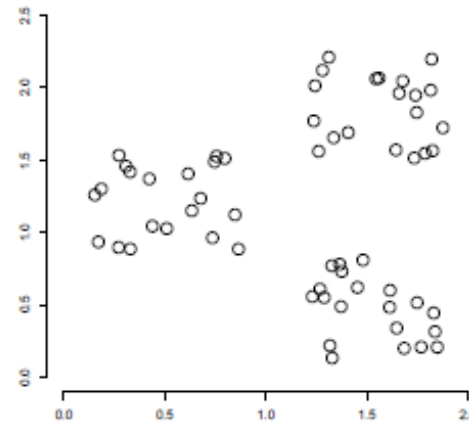
---

# 16

*Flat clustering*

## CLUSTER

Clustering algorithms group a set of documents into subsets or *clusters*. The algorithms' goal is to create clusters that are coherent internally, but clearly different from each other. In other words, documents within a cluster should be as similar as possible; and documents in one cluster should be as dissimilar as possible from documents in other clusters.



► **Figure 16.1** An example of a data set with a clear cluster structure.

UNSUPERVISED  
LEARNING

Clustering is the most common form of *unsupervised learning*. No supervision means that there is no human expert who has assigned documents to classes. In clustering, it is the distribution and makeup of the data that will determine cluster membership. A simple example is Figure 16.1. It is

# Distance Measure Determines Clustering

The key input to a clustering algorithm is the distance measure. In Figure 16.1, the distance measure is distance in the 2D plane. This measure suggests three different clusters in the figure. In document clustering, the distance measure is often also Euclidean distance. Different distance measures give rise to different clusterings. Thus, the distance measure is an important means by which we can influence the outcome of clustering.



## 16.1 Clustering in information retrieval

CLUSTER HYPOTHESIS    The *cluster hypothesis* states the fundamental assumption we make when using clustering in information retrieval.

**Cluster hypothesis.** Documents in the same cluster behave similarly with respect to relevance to information needs.

PARTITIONAL  
CLUSTERING

**A note on terminology.** An alternative definition of hard clustering is that a document can be a full member of more than one cluster. *Partitional clustering* always refers to a clustering where each document belongs to exactly one cluster. (But in a partitional hierarchical clustering (Chapter 17) all members of a cluster are of course also members of its parent.) On the definition of hard clustering that permits multiple membership, the difference between soft clustering and hard clustering is that membership values in hard clustering are either 0 or 1, whereas they can take on any non-negative value in soft clustering.

EXHAUSTIVE

Some researchers distinguish between *exhaustive* clusterings that assign each document to a cluster and non-exhaustive clusterings, in which some documents will be assigned to no cluster. Non-exhaustive clusterings in which each document is a member of either no cluster or one cluster are called *exclusive*. We define clustering to be exhaustive in this book.

EXCLUSIVE

## 16.4 *K*-means

*K*-means is the most important flat clustering algorithm. Its objective is to minimize the average squared Euclidean distance (Chapter 6, page 131) of documents from their cluster centers where a cluster center is defined as the mean or *centroid*  $\vec{\mu}$  of the documents in a cluster  $\omega$ :

CENTROID

$$\vec{\mu}(\omega) = \frac{1}{|\omega|} \sum_{\vec{x} \in \omega} \vec{x}$$

RESIDUAL SUM OF  
SQUARES

A measure of how well the centroids represent the members of their clusters is the *residual sum of squares* or *RSS*, the squared distance of each vector from its centroid summed over all vectors:

$$\text{RSS}_k = \sum_{\vec{x} \in \omega_k} |\vec{x} - \vec{\mu}(\omega_k)|^2$$

(16.7)

$$\text{RSS} = \sum_{k=1}^K \text{RSS}_k$$

RSS is the objective function in  $K$ -means and our goal is to minimize it. Since  $N$  is fixed, minimizing RSS is equivalent to minimizing the average squared distance, a measure of how well centroids represent their documents.

# K-Means

- Assumes documents are real-valued vectors.
- Clusters based on *centroids* (aka the *center of gravity* or mean) of points in a cluster,  $c$ :

$$\vec{\mu}(c) = \frac{1}{|c|} \sum_{\vec{x} \in c} \vec{x}$$

- Reassignment of instances to clusters is based on distance to the current cluster centroids.
  - (Or one can equivalently phrase it in terms of similarities)

# K-Means Algorithm

Select  $K$  random docs  $\{s_1, s_2, \dots, s_K\}$  as seeds.

Until clustering *converges* (or other stopping criterion):

For each doc  $d_i$ :

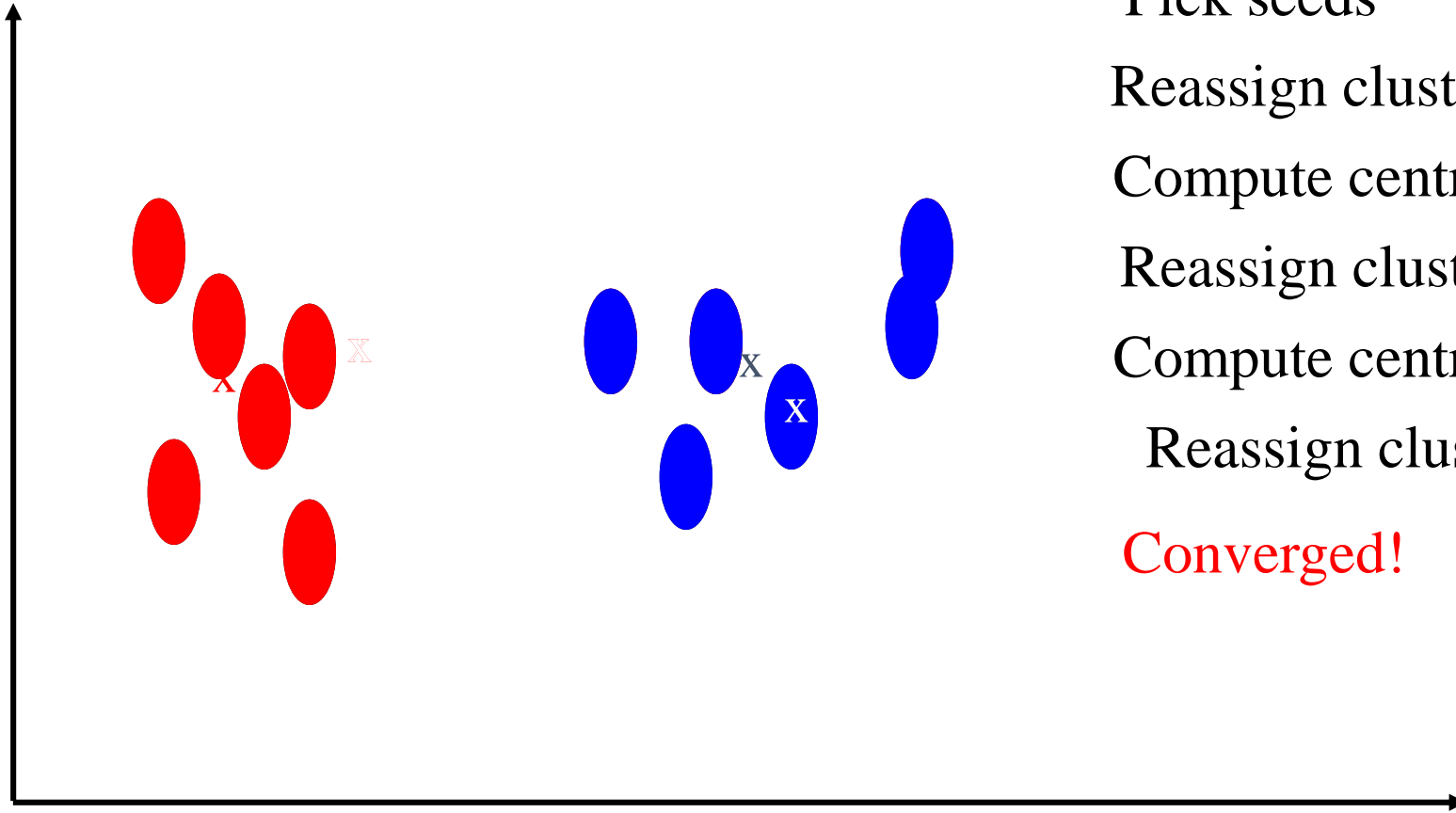
Assign  $d_i$  to the cluster  $c_j$  such that  $\text{dist}(x_i, s_j)$  is minimal.

*(Next, update the seeds to the centroid of each cluster)*

For each cluster  $c_j$

$$s_j = \mu(c_j)$$

# K Means Example ( $K=2$ )



Pick seeds

Reassign clusters

Compute centroids

Reassign clusters

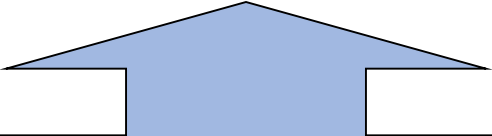
Compute centroids

Reassign clusters

**Converged!**

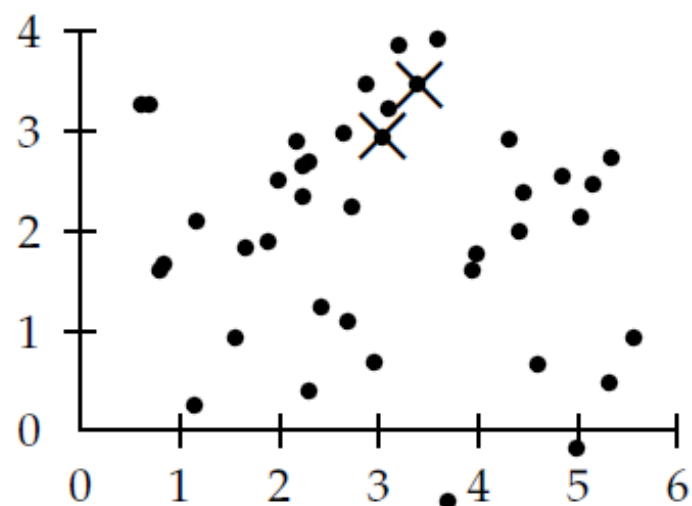
# Termination conditions

- Several possibilities, e.g.,
  - A fixed number of iterations.
  - Doc partition unchanged.
  - Centroid positions don't change.

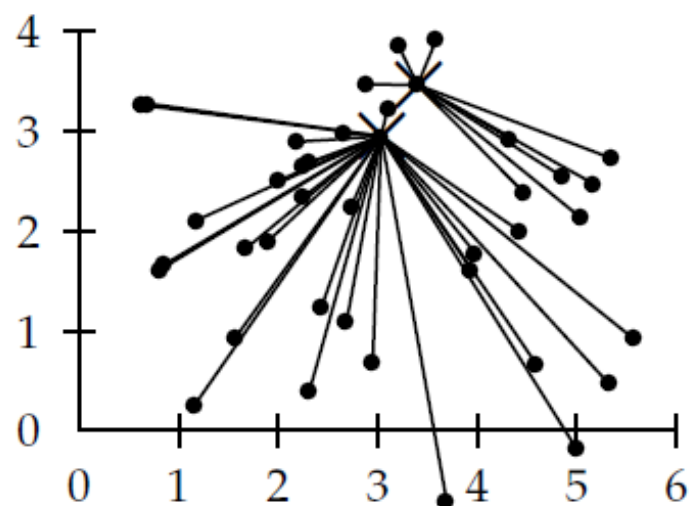


Does this mean that the docs in a cluster are unchanged?

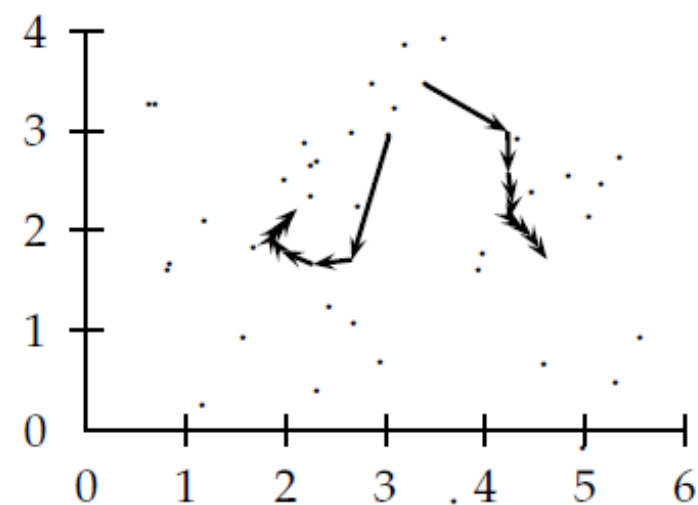




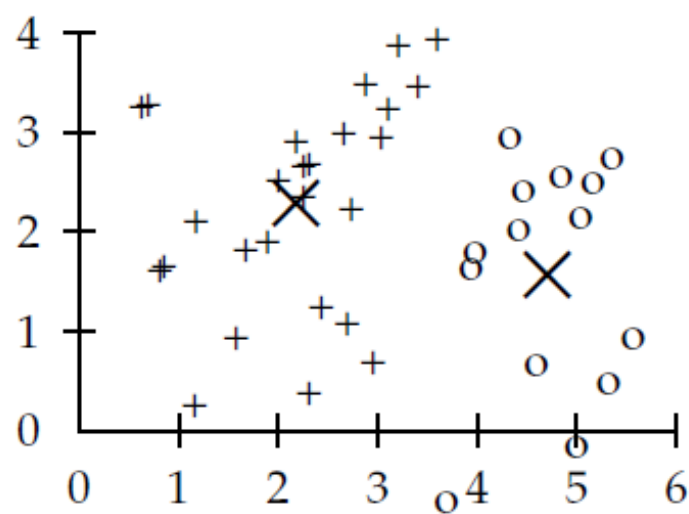
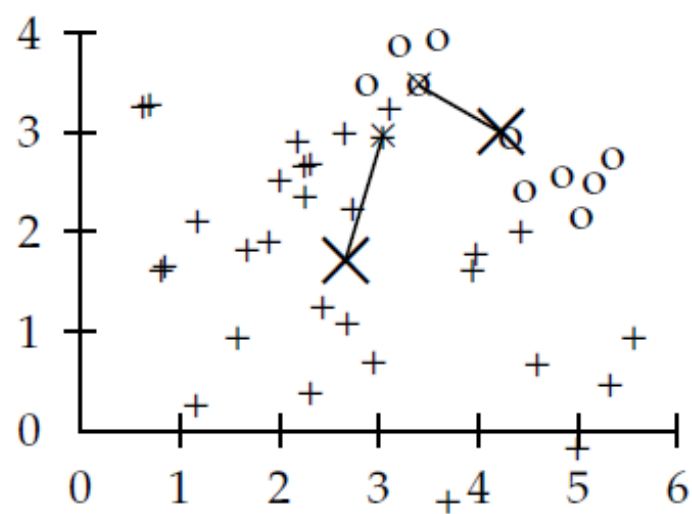
selection of seeds



assignment of documents (iter. 1)



movement of  $\bar{\mu}'$ 's in 9 iterations



recomputation/movement of  $\bar{\mu}'$ 's (iter. 1)     $\bar{\mu}'$ 's after convergence (iter. 9)

► Figure 16.6 A  $K$ -means example for  $K = 2$  in  $\mathbb{R}^2$ .

# RSS Decreases $\rightarrow$ K-Means Converges

We now show that  $K$ -means converges by proving that RSS monotonically decreases in each iteration. We will use *decrease* in the meaning *decrease or does not change* in this section. First, RSS decreases in the reassignment step since each vector is assigned to the closest centroid, so the distance it contributes to RSS decreases. Second, it decreases in the recomputation step because the new centroid is the vector  $\vec{v}$  for which  $\text{RSS}_k$  reaches its minimum.

$$(16.8) \quad \text{RSS}_k(\vec{v}) = \sum_{\vec{x} \in \omega_k} |\vec{v} - \vec{x}|^2 = \sum_{\vec{x} \in \omega_k} \sum_{m=1}^M (v_m - x_m)^2$$

# Mean Minimizes RSS

$$(16.9) \quad \frac{\partial \text{RSS}_k(\vec{v})}{\partial v_m} = \sum_{\vec{x} \in \omega_k} 2(v_m - x_m)$$

where  $x_m$  and  $v_m$  are the  $m^{\text{th}}$  components of their respective vectors. Setting the partial derivative to zero, we get:

$$(16.10) \quad v_m = \frac{1}{|\omega_k|} \sum_{\vec{x} \in \omega_k} x_m$$

which is the componentwise definition of the centroid. Thus, we minimize  $\text{RSS}_k$  when the old centroid is replaced with the new centroid. RSS, the sum of the  $\text{RSS}_k$ , must then also decrease during recomputation.

# No Guarantee to a Global Minimum

While this proves the convergence of  $K$ -means, there is unfortunately no guarantee that a *global minimum* in the objective function will be reached. This is a particular problem if a document set contains many *outliers*, documents that are far from any other documents and therefore do not fit well into any cluster. Frequently, if an outlier is chosen as an initial seed, then no other vector is assigned to it during subsequent iterations. Thus, we end up with a *singleton cluster* (a cluster with only one document) even though there is probably a clustering with lower RSS. Figure 16.7 shows an example of a suboptimal clustering resulting from a bad choice of initial seeds.

Another type of suboptimal clustering that frequently occurs is one with empty clusters (Exercise 16.11).

```
# IR18A.py CS5154/6054 cheng 2022
# k-means by TrainRocchio and ApplyRocchio
# Usage: python IR18A.py
```

```
import numpy as np
import random
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.neighbors import NearestCentroid
from matplotlib import pyplot as plt
from sklearn.metrics import accuracy_score
```

```
f = open("bible.txt", "r")
docs = f.readlines()
f.close()
N = len(docs)
K = 8
```

```
    RSS = 0
    for i in range(N):
        d = X[i] - model.centroids_[y[i]]
        RSS += np.dot(d, d.T).item()
    print(RSS)
```


```
cv = TfidfVectorizer(max_df=0.4, min_df=4)
X = cv.fit_transform(docs)
seeds = random.sample(range(N), K)
centroids = X[seeds]
classes = list(range(K))
init = NearestCentroid()
init.fit(centroids, classes)
y = init.predict(X)
plt.hist(y)
plt.show()
```

```
model = NearestCentroid()
for iter in range(10):
    model.fit(X, y)
    pred = model.predict(X)
    print(accuracy_score(y, pred))
    y = pred
    plt.hist(y)
    plt.show()
```

FileEditViewHistoryBookmarksToolsHelp

sklearn.cluster.KMeans — scikit-learn

←→↻🏠🔒https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans📄150%★📧📌🔍☰



☰

# sklearn.cluster.KMeans

```
class sklearn.cluster.KMeans(n_clusters=8, *, init='k-means++', n_init=10,
max_iter=300, tol=0.0001, verbose=0, random_state=None, copy_x=True,
algorithm='lloyd')
```

[\[source\]](#)

K-Means clustering.

Toggle Menu

[the User Guide.](#)

```
# IR18B.py CS5154/6054 cheng 2022
```

```
# k-means
```

```
# Usage: python IR18B.py
```

```
import numpy as np
```

```
import random
```

```
from sklearn.feature_extraction.text import TfidfVectorizer
```

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

```
f = open("bible.txt", "r")
```

```
docs = f.readlines()
```

```
f.close()
```

```
N = len(docs)
```

```
cv = TfidfVectorizer(max_df=0.4, min_df=4)
```

```
X = cv.fit_transform(docs)
```

```
model = KMeans(n_init=1, max_iter=10)
```

```
model.fit_predict(X)
```

```
y = model.labels_
```

```
RSS = 0
```

```
for i in range(N):
```

```
    d = X[i] - model.cluster_centers_[y[i]]
```

```
    RSS += np.dot(d, d.T).item()
```

```
print(RSS)
```