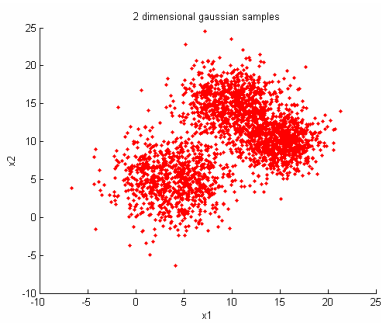


K-Means & K Nearest Neighbors

Chen Yu
Indiana University

K-Means



K-Means

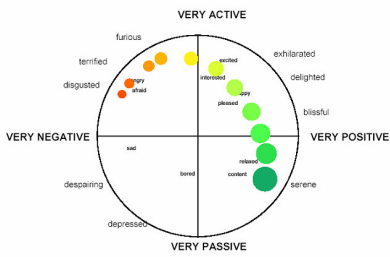
- The algorithm can group your data into k number of categories.
- The principle is to minimize the sum of squares of distances between data and the corresponding cluster centroids.

Example

arousal(-5 to 5)	valance
3	3
-1	-4
2	3
0	-5

We know that those data belong to two clusters.
The question is how to determine which data points belong to cluster 1 and which belong to the other one.

Example



K-Mean Algorithm

- Repeat the following three steps until convergence (stable):
- Step 1: determine the centroid coordinates
 - Step 2: determine the distances of each data point to the centroids.
 - Step 3: group the data points based on minimum distance.

Example

Initialize the first two centroids

$c1=(3,3)$ $c2=(2,3)$

3	3
-1	-4
2	3
0	-5

Measuring distances

- Calculate the distance between two data items.
- Euclidean distance

$$\sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \dots + (p_n - q_n)^2} = \sqrt{\sum_{i=1}^n (p_i - q_i)^2}$$

Iteration 1: calculate distances

		$c1=(3,3)$	$c2=(2,3)$
3	3	0	1
-1	-4	8.1	7.6
2	3	1	0
0	-5	8.5	8.2

Iteration 1: assign clusters

		$c1=(3,3)$	$c2=(2,3)$
3	3	0	1
-1	-4	8.1	7.6
2	3	1	0
0	-5	8.5	8.2

Iteration 1: compute new centroids

		$c1=(3,3)$	$c2=(0.3,-2)$
3	3	0	1
-1	-4	8.1	7.6
2	3	1	0
0	-5	8.5	8.2

Iteration 2: calculate distances

		$c1=(3,3)$	$c2=(0.3,-2)$
3	3	0	5.7
-1	-4	8.1	2.4
2	3	1	5.3
0	-5	8.5	3.0

Iteration 2: assign clusters

		$c1=(3,3)$	$c2=(0.3,-2)$	
3	3	0	5.7	
-1	-4	8.1	2.4	
2	3	1	5.3	
0	-5	8.5	3.0	

Iteration 2: compute new centroids

		$c1=(2.5,3)$	$c2=(-0.5,-4.5)$	
3	3	0	5.7	
-1	-4	8.1	2.4	
2	3	1	5.3	
0	-5	8.5	3.0	

Iteration 3

		$c1=(2.5,3)$	$c2=(-0.5,-4.5)$	
3	3	0.5	8.2	
-1	-4	7.8	0.7	
2	3	0.5	7.9	
0	-5	8.3	0.7	

The new centroids will be the same!

```
function c = kmean(k,data)
```

```
[ndat ndim] = size(data);
```

```
% initialize the centra point  
r = randperm(ndat);  
c(1:k,:) = data(r(1:k),:);
```

```
ctemp = zeros(size(c));  
cluster = zeros(size(data,1));  
it = 0;
```

```
while (c ~= ctemp)
```

```
it = it + 1;
```

```
fprintf(1,'iteration %d: (%6.3f,%6.3f),(%6.3f,%6.3f),(%6.3f,%6.3f)\n', it,  
c(1,1),c(1,2),c(2,1),c(2,2),c(3,1),c(3,2));  
plot(data(:,1),data(:,2),'r',c(:,1),c(:,2),'b')  
pause;
```

```
ctemp = c;  
dist = distance(data,c);  
[non cluster] = min(dist,[],2);  
for i = 1 : k  
c(i,:) = mean(data(find(cluster == i,:)));  
end;  
end;
```

Matlab Example

Classification

- Supervised: given training samples, classify a new instance.
- Features: attributes of an object that can be represented a multidimensional space. e.g. object categorization, face recognition, emotion recognition, etc.

K Nearest Neighbors

- The classification is using majority vote.
- The classifiers do not use any model to fit the data and only based on memory.
- The KNN uses neighborhood classification as the predication value of the new query instance.

Example

Two features:

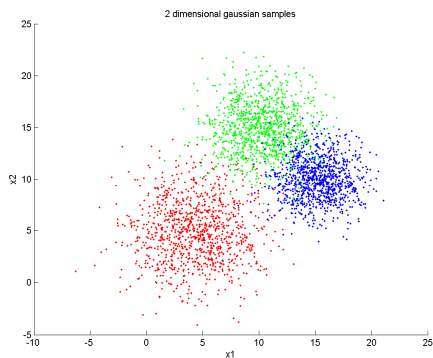
arousal(-5 to 5)	valance	emotion
3	3	happy
-1	-4	sad
2	3	happy
0	-5	sad

Now given a new instance (3,4), what category is this new data point in?

How KNN works?

- The general principle is to find the k training samples to determine the k-nearest neighbors based on a distance measure. Next, the majority of those k nearest neighbors decide the category of the next instance.

Example



Finding K-nearest neighbors

- Sort the distances of all training samples to the new instance and determine the k-th minimum distance.

Classification

- Find the majority of the category of k nearest neighbors.

Summary

- Step 1: determine k
- Step 2: calculate the distances between the new input and all the training data
- Step 3: sort the distance and determine k nearest neighbors based on the k-th minimum distance.
- Step 4: gather the categories of those neighbors.
- Step 5: determine the category based on majority vote.

Example

arousal(-5 to 5)	valance	emotion
3	3	happy
-1	-4	sad
2	3	happy
0	-5	sad

(3,4) = ?

Example

- Step 1: $k = 3$

• Step 2:			distance
3	3		1
-1	-4	to (3,4)	8.9
2	3		1.4
0	-5		9.4

Step 3: ranking

		rank	neighbors
3	3	1	Y
2	3	2	Y
-1	-4	3	Y
0	-5	4	N

Step 4: categorization

		rank	neighbors	category
3	3	1	Y	happy
2	3	2	Y	happy
-1	-4	3	Y	sad
0	-5	4	N	sad
