UNSUPERVISED K-MEANS CLUSTERING ALGORITHM

Mohammad Shoaei

Feb. 2022

TABLE OF CONTENTS

CLUSTERING APPROACHES

CLUSTERING APPROACHES

1. Probability model-based
    - Mixture models
2. Nonparametric
    - Hierarchical
    - Partitional

UNSUPERVISED K-MEANS

UNSUPERVISED K-MEANS

- Is K-Means algorithm a true unsupervised algorithm?
- What is the major problem of K-Means algorithm and its extension?

UNSUPERVISED K-MEANS

- X-Means
  - an algorithm to estimate the number of clusters
- Unsupervised K-Means Clustering Algorithm (U-K-Means)
  - New proposed algorithm which finds the number of clusters automatically

U-K-MEANS

Definitions:

1. Z: membership matrix $\epsilon\ R^{n*\mathrm{c}}$ where $\mathrm{Z[i,k]}$ is 1 if sample x_i belongs to cluster k
2. α: a list where each element k determines the probability that a data point belongs to cluster k
3. a: list of current centroids
4. Entropy: $\Sigma_{k=1}^{c}\alpha_k \ln \alpha_k$
5. β: parameter to control the competition
6. γ: learning parameter

U-K-MEANS

Objective Function:

$$J_{U-k-means}(\mathrm{z}, A, \alpha) = \sum_{i=1}^{n}\sum_{k=1}^{c} z_{ik} \|x_i - a_k\|^2 - \beta n \sum_{k=1}^{c} \alpha_k \ln \alpha_k - \gamma \sum_{i=1}^{n}\sum_{k=1}^{c} z_{ik} \ln \alpha_k \quad (2)$$

U-K-MEANS

Updating the probabilities:

$$\alpha_k^{(t+1)} = \sum_{i=1}^{n} z_{ik}/n + (\beta/\gamma)\alpha_k^{(t)} \left(\ln \alpha_k^{(t)} - \sum_{s=1}^{c} \alpha_s^{(t)} \ln \alpha_s^{(t)} \right) \tag{6}$$

U-K-MEANS

U-k-means clustering algorithm

Step 1: Fix $\varepsilon > 0$. Give initial $c^{(0)} = n$, $\alpha_k^{(0)} = 1/n$, $a_k^{(0)} = x_i$, and initial learning rates $\gamma^{(0)} = \beta^{(0)} = 1$. Set $t = 0$.

Step 2: Compute $z_{ik}^{(t+1)}$ using $a_k^{(t)}, \alpha_k^{(t)}, c^{(t)}, \gamma^{(t)}, \beta^{(t)}$ by (4).

Step 3: Compute $\gamma^{(t+1)}$ by (10).

Step 4: Update $\alpha_k^{(t+1)}$ with $z_{ik}^{(t+1)}$ and $\alpha_k^{(t)}$ by (6).

Step 5: Compute $\beta^{(t+1)}$ with $\alpha^{(t+1)}$ and $\alpha^{(t)}$ by (14).

Step 6: Update $c^{(t)}$ to $c^{(t+1)}$ by discard those clusters with $\alpha_k^{(t+1)} \leq 1/n$ and adjust $\alpha_k^{(t+1)}$ and $z_{ik}^{(t+1)}$ by (8) and (9).
IF $t \geq 60$ and $c^{(t-60)} - c^{(t)} = 0$, THEN let $\beta^{(t+1)} = 0$.

Step 7: Update $a_k^{(t+1)}$ with $c^{(t+1)}$ and $z_{ik}^{(t+1)}$ by (5).

Step 8: Compare $a_k^{(t+1)}$ and $a_k^{(t)}$.
IF $\max\limits_{1 \leq k \leq c^{(t)}} \left\| a_k^{(t+1)} - a_k^{(t)} \right\| < \varepsilon$, THEN Stop.
ELSE $t = t+1$ and return to Step 2.

U-K-MEANS

Notable problems of the paper:

1. Matrix Z will be an identity matrix on 0^{th} iteration
2. Recall that it is not mentioned which β and γ should be used to update α
3. Inconsistent criteria for updating the number of clusters
4. Despite strictly defining that matrix Z has either value 1 or 0 but algorithm (9) makes this matrix contain other values than 0 or 1.

$$z_{ik}^{*} = z_{ik}^{*} \Big/ \sum_{s=1}^{c^{(t+1)}} z_{is}^{*} \qquad (9)$$

Hello Shoaei. Sorry for the late reply.

I reviewed your code and paper carefully and I did simple experiment with your code.
You are right. Matrix Z on iteration 0 would be an identity matrix and it causes no updated for alpha.
Because there are no update for alpha, algorithm does not work.

IMPLEMENTATION OF U-K-MEANS

IMPLEMENTATION

The Algorithm is defined as a Python class described below

```
class UKMeans(epsilon=1e-5)
```

IMPLEMENTATION

Methods:

```python
def _compute_z(self, X: np.ndarray)
    "compute the matrix Z"

def _update_gamma(self)
    "update gamma using algorithm (10)"

def _update_alpha(self, X: np.ndarray)
    "update each alpha using algorithm (6)"

def _update_beta(self, X: np.ndarray, alpha_t: np.ndarray)
    """update beta using algorithm (14).
    current alpha array must be passed"""

def _update_c_alpha_z(self, X: np.ndarray) -> np.ndarray:
    """this is step 6 of the proposed algorithm
    which updates the number of clusters"""

def _update_a(self, X: np.ndarray)
    "update centroids using algorithm (5)"
```

IMPLEMENTATION

Methods:

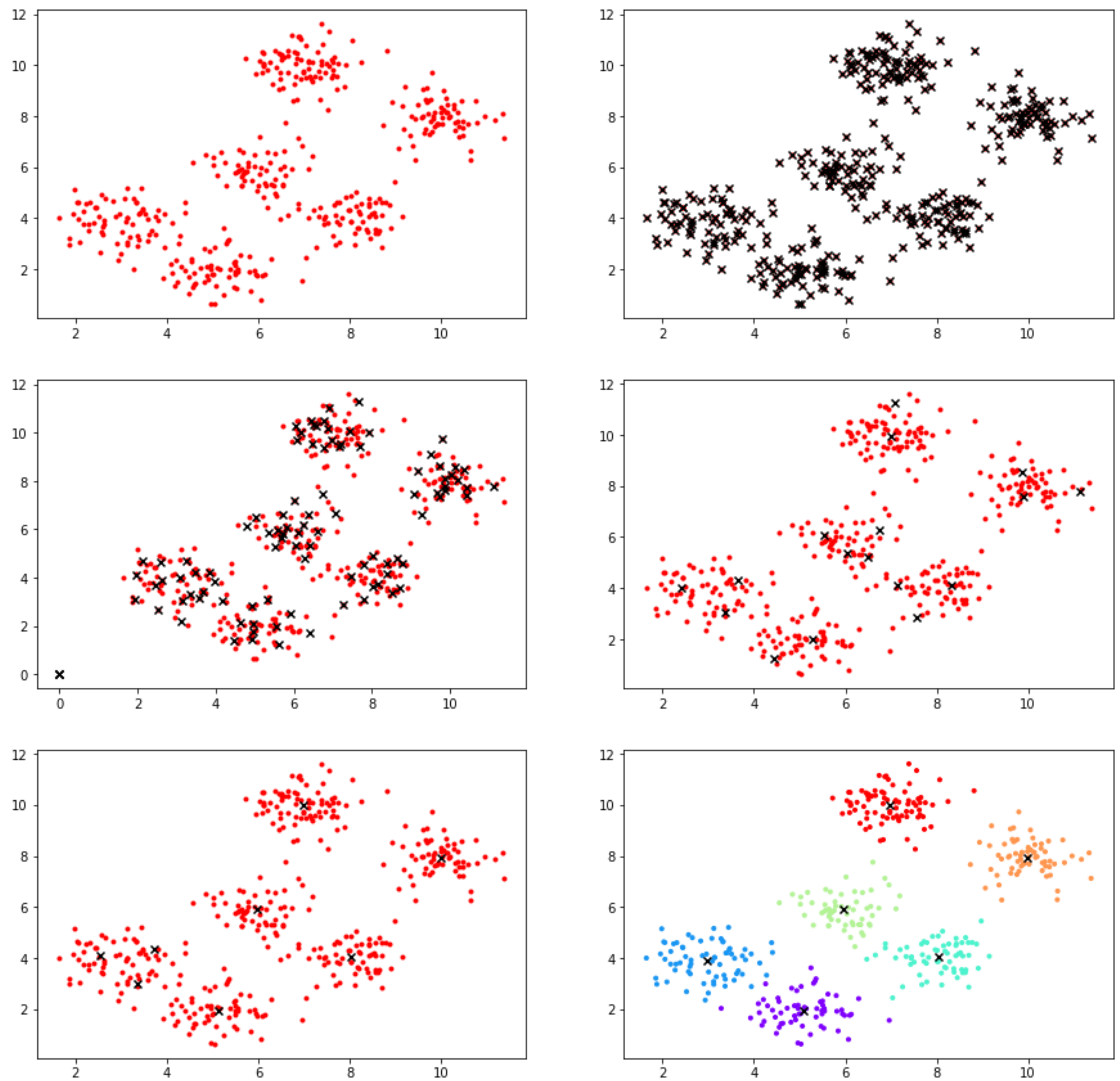
```
def fit(self, X: np.ndarray)
    "main method which must be called to fit the model"

def predict(self, X: np.ndarray)
    """predict the labels of the data
    the lables will differ from the original labels
    because the algorithm is not deterministic"""

def accuracy_rate(self, X: np.ndarray, y_true: np.ndarray)
    """if the number of clusters is found correctly
    then this method finds the original label of each sample
```

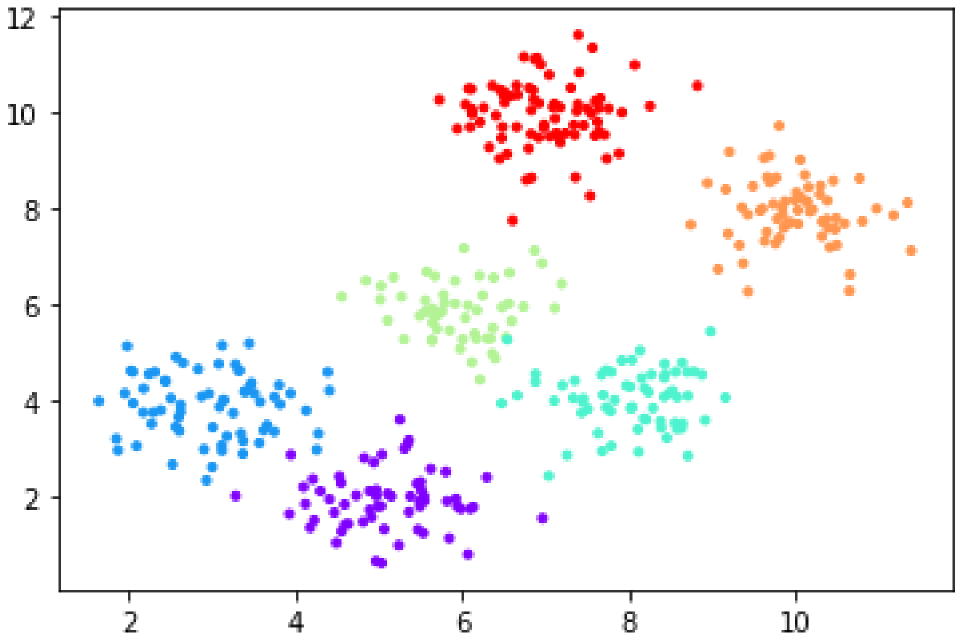
EXPERIMENTAL RESULTS

EXPERIMENTAL RESULTS EXAMPLE 1

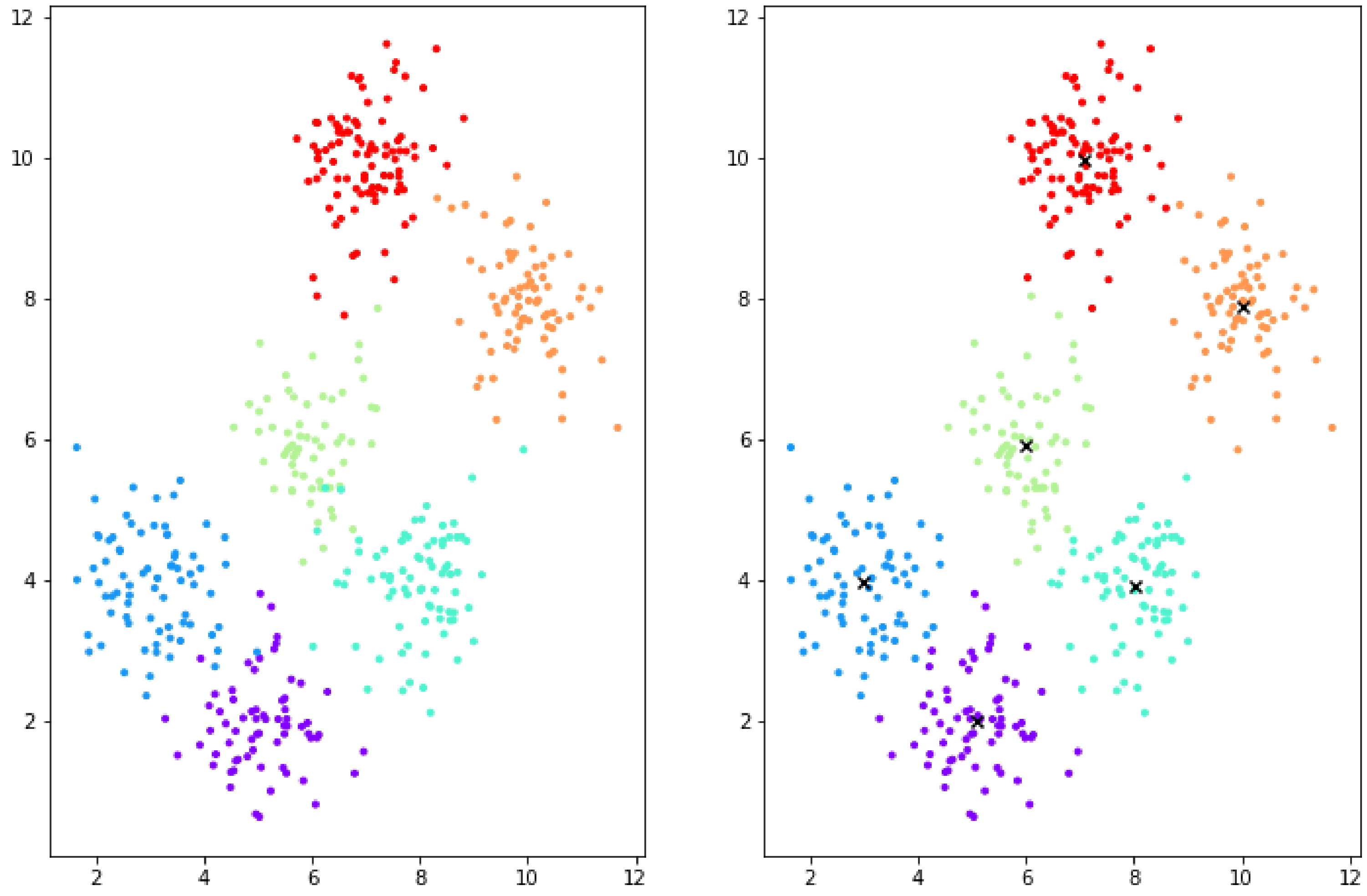


Without Noise
- Iteration = 13
- C* = 6
- AR = 0.99

With Noise
- Iteration = 20
- C* = 6
- AR = 0.968



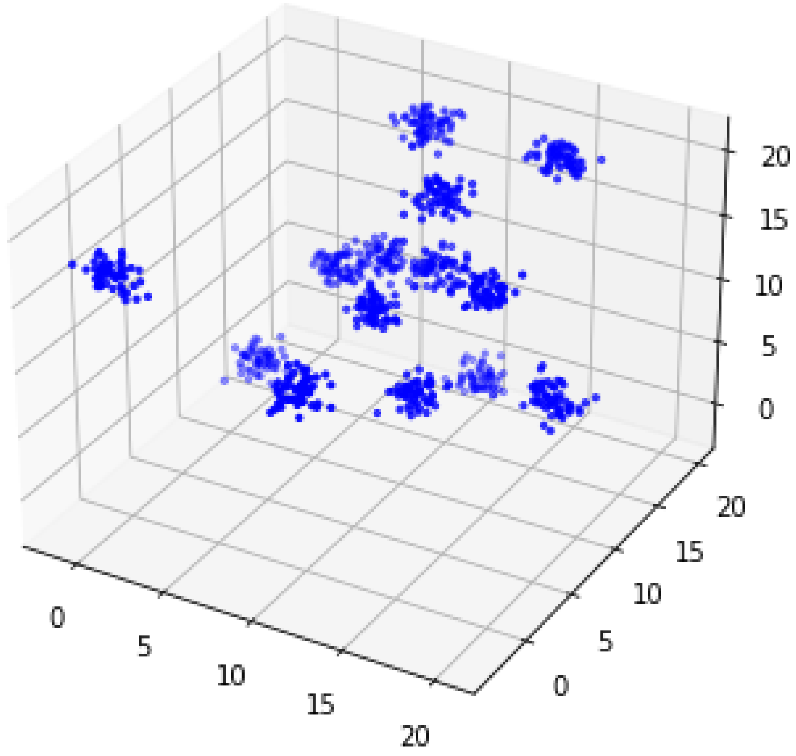
EXPERIMENTAL RESULTS EXAMPLE 1

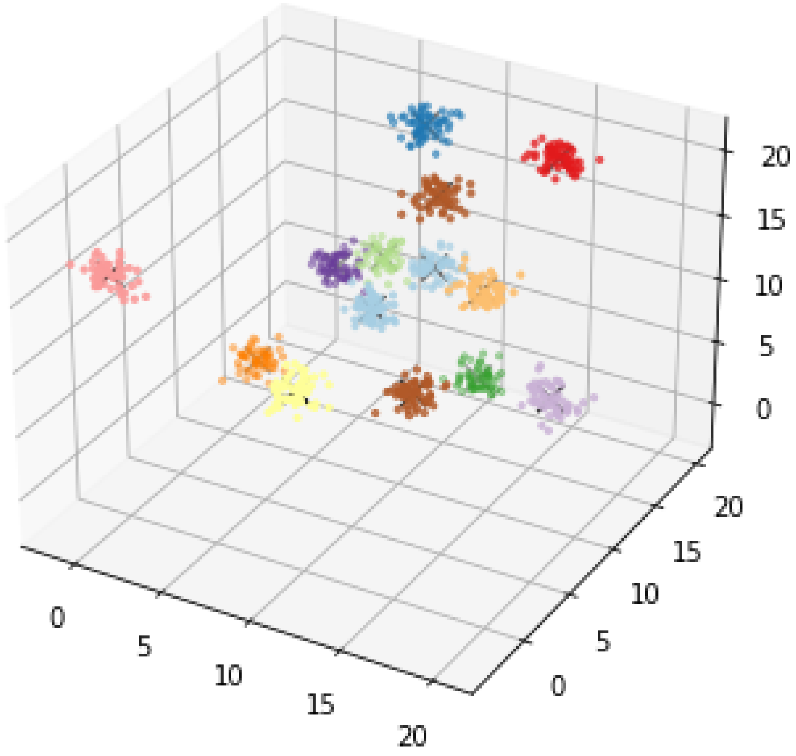


EXPERIMENTAL RESULTS EXAMPLE 2

3-variate, 14-components

- Iteration = 20
- C* = 14
- AR = 0.99625





EXPERIMENTAL RESULTS EXAMPLE 3

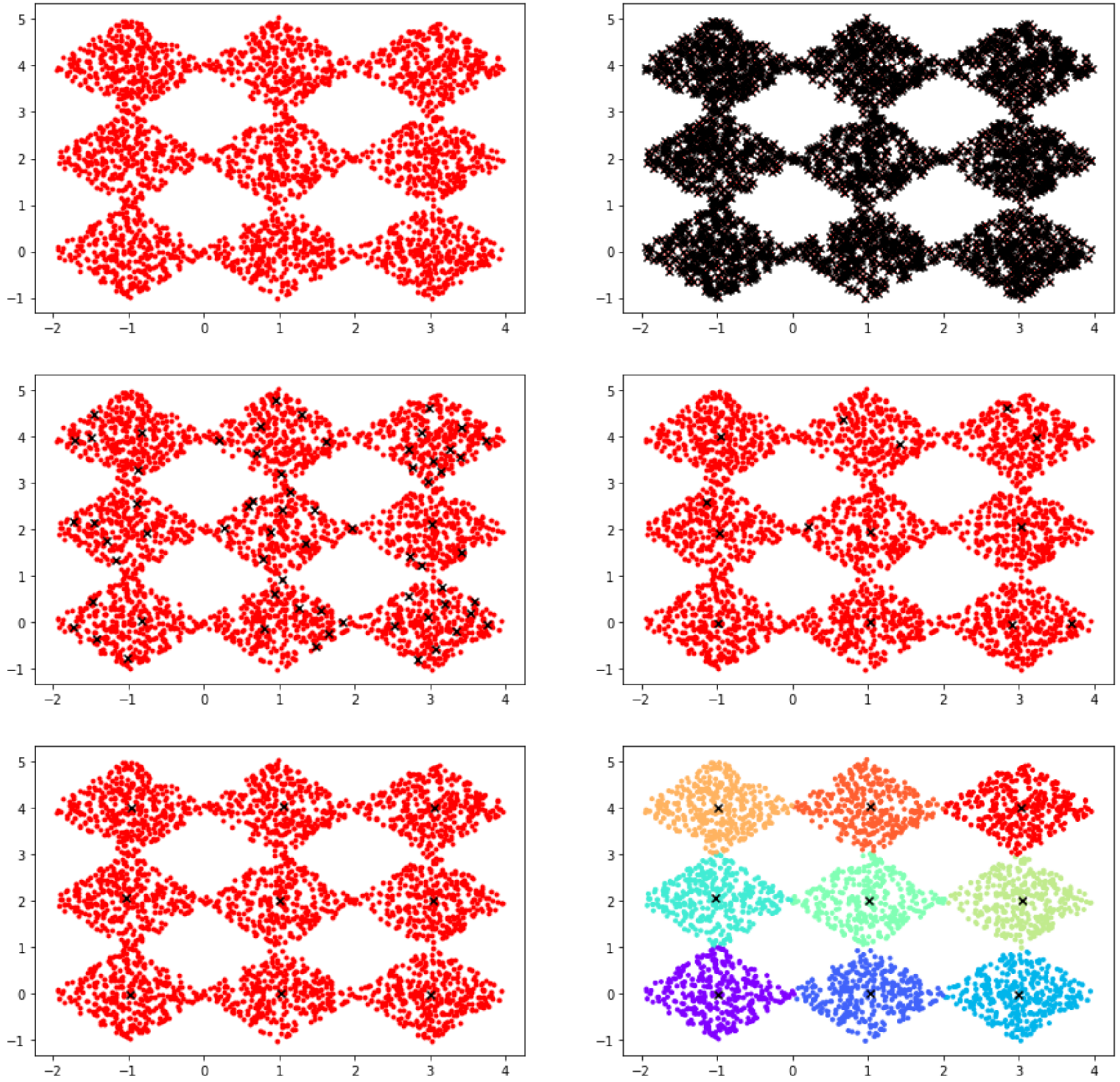
Mixing proportions	Mean values	covariance matrix
$\alpha_1 = 0.2$	$\mu_1 = (2 \quad 4 \quad 6 \quad 0 \quad 0 \quad 0 \quad 0 \quad 1 \quad 1 \quad 1 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 3 \quad 5 \quad 0 \quad 0 \quad 1)$	$\sum_k = I_{[20\times20]}$
$\alpha_2 = 0.3$	$\mu_2 = (0 \quad 1 \quad 3 \quad 5 \quad 0.1 \quad 0.1 \quad 0.5 \quad 0.5 \quad 0 \quad 0 \quad 2 \quad 4 \quad 3 \quad 1 \quad 1 \quad 1 \quad 0.25 \quad 0.5 \quad 0.7 \quad 2.5)$	
$\alpha_3 = 0.1$	$\mu_3 = (5 \quad 5 \quad 5 \quad 5 \quad 4 \quad 4 \quad 4 \quad 4 \quad 6 \quad 6 \quad 6 \quad 6 \quad 8 \quad 8 \quad 8 \quad 8 \quad 1 \quad 1 \quad 1 \quad 1)$	
$\alpha_4 = 0.1$	$\mu_4 = (2 \quad 2 \quad 2 \quad 2 \quad 2 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 3 \quad 3 \quad 3 \quad 3 \quad 3 \quad 7 \quad 7 \quad 7 \quad 7 \quad 7)$	
$\alpha_5 = 0.2$	$\mu_5 = (1.25 \quad 1.3 \quad 1.45 \quad 1.5 \quad 2.25 \quad 2.3 \quad 2.45 \quad 2.5 \quad 1 \quad 1 \quad 1 \quad 1 \quad 3 \quad 3 \quad 3 \quad 3 \quad 2 \quad 2 \quad 2 \quad 2)$	
$\alpha_6 = 0.1$	$\mu_6 = (0 \quad 0 \quad 1 \quad 1 \quad 0.5 \quad 0.5 \quad 2.5 \quad 2.5 \quad 5 \quad 5 \quad 1 \quad 1 \quad 5 \quad 5 \quad 0 \quad 0 \quad 0.75 \quad 1.5 \quad 3.5 \quad 5.5)$	

20-variate, 6-components
900 samples

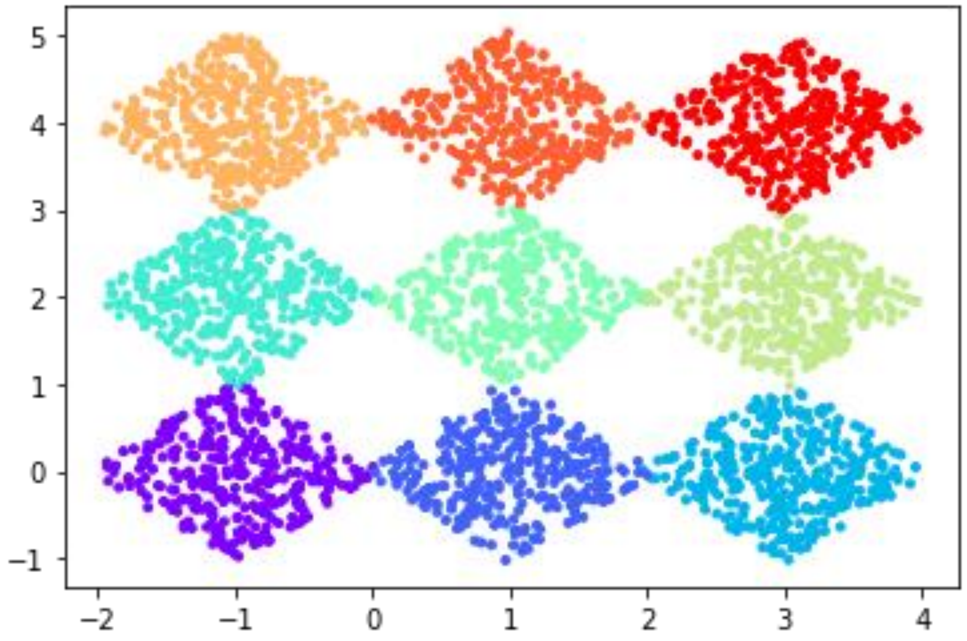
Result:

- Iteration = 11
- C* = 6
- AR = 1.0

EXPERIMENTAL RESULTS EXAMPLE 4



- Iteration = 11
- C* = 9
- AR = 0.996



EXPERIMENTAL RESULTS EXAMPLE 5

Dataset	True C	C*	AR
Iris	3	3	0.8866
Seeds	3	3	0.8857
Australian	2	1	-
Flowmeter D	4	2	-
Sonar (*)			
Wine	3	3	0.6067
Horse	2	1	-
Waveform V1	3	4	-

*: dataset was not found in UCI repository

EXPERIMENTAL RESULTS EXAMPLE 6

Dataset	True C	C*	AR
SPECT	2	2	0.4385
PARKINSON	2	2	0.6974
WPBC	2	2	0.8541
COLON(*)			
LUNG (*)			
Nci9(*)			

*: dataset was not found in UCI repository

EXPERIMENTAL RESULTS EXAMPLE 7

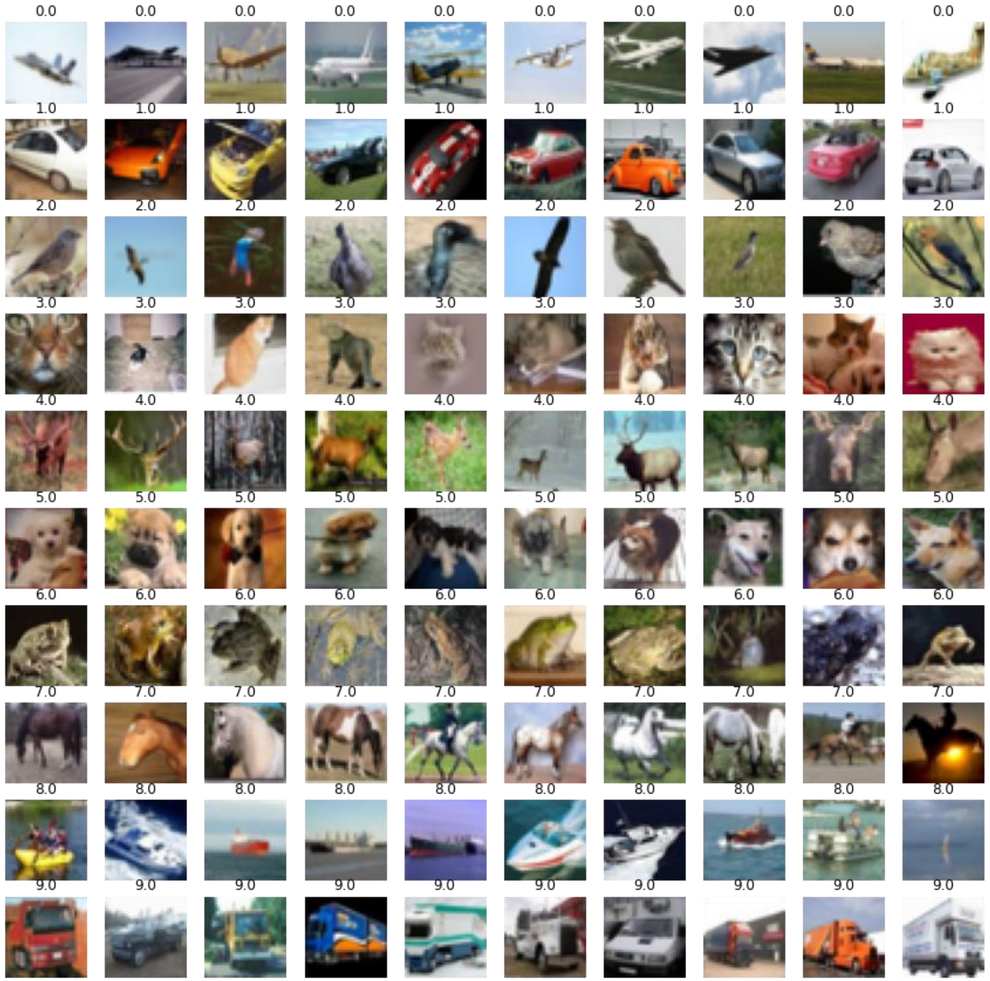
- It is unclear how 135 out of 165 images were selected.
- The result in the following table are using the first 9 images of each person.
- Tweaking the number of images changed the number of clusters with a maximum 6 cluster for selecting the first 7

Dataset	True C	C*	AR	Paper C*
Yale Face 32x32	15	2	-	16

EXPERIMENTAL RESULTS EXAMPLE 8

Dataset	True C	C*	AR	Paper C
CIFAR-10	10	8	-	10

- The paper does display the sample that were chosen from the dataset
- It is stated that there are 10 image per class
- The image displaying the samples used in paper contains 11 pictures from horse class which affects the authenticity of results achieved by the paper



EXPERIMENTAL RESULTS PERFORMANCE

Dataset	Time (s)
Synthetic Dataset	
Example 1	2.49
Example 2	9.07
Example 3	10.3
Example 4	74

Dataset		Dataset	Time (s)
UCI Data		UCI Dataset	
Iris		Iris	0.251
Seeds		Seeds	0.682
Australian		Australian	6.31
Flowmeter D		Flowmeter D	0.923
Sonar(*)		Sonar(*)	
Wine		Wine	0.557
Horse(*)		Horse(*)	
Waveform		Waveform	240

EXPERIMENTAL RESULTS PERFORMANCE

Dataset	Time (s)
Medical Dataset	
SPECT	0.440
Parkinson	1.08
WPBC	5.99
Colon(*)	
LUNG(*)	
Nci9(*)	

Dataset	Time (s)
Image Dataset	
Yale Face 32x32	0.349
CIFAR-10	0.218