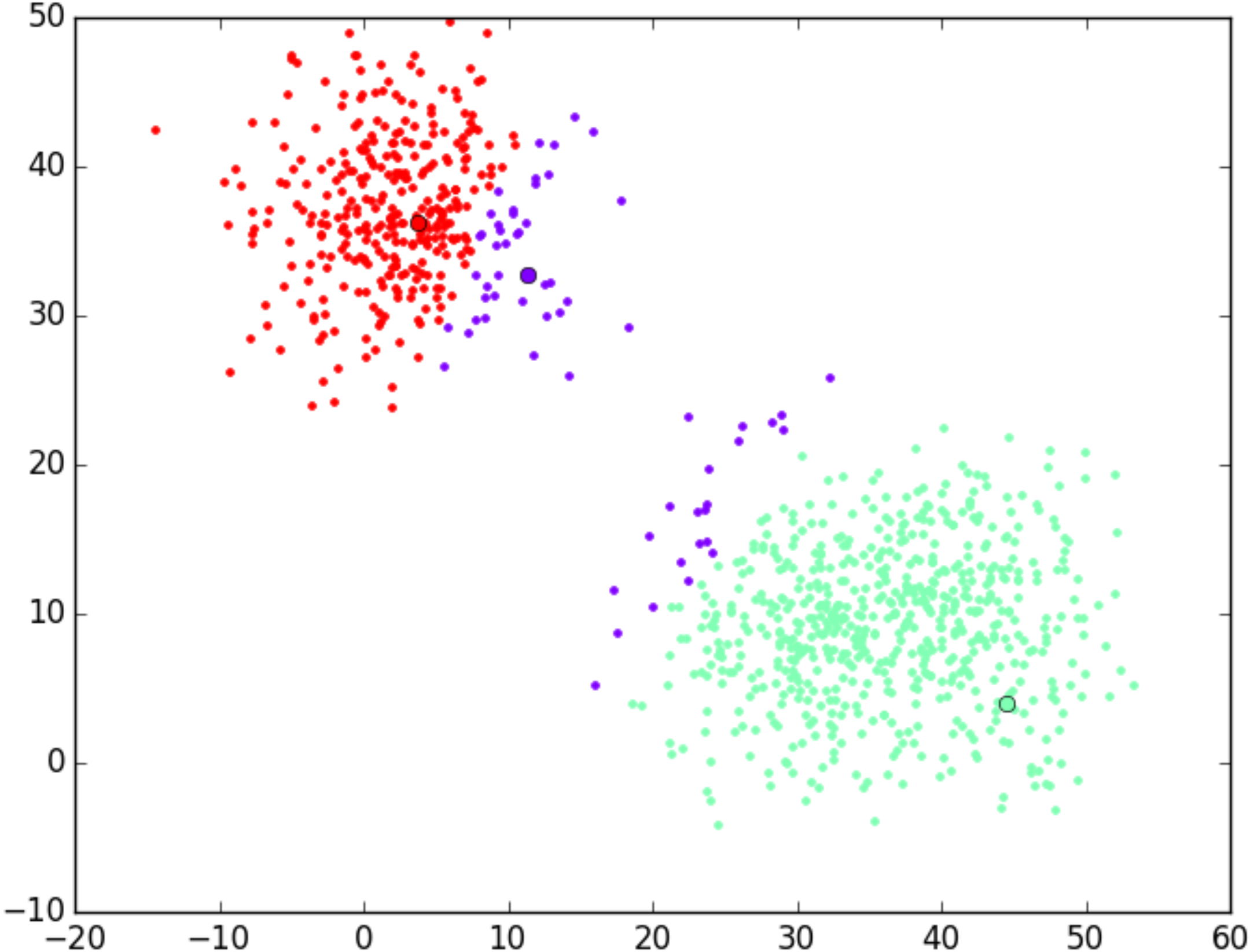
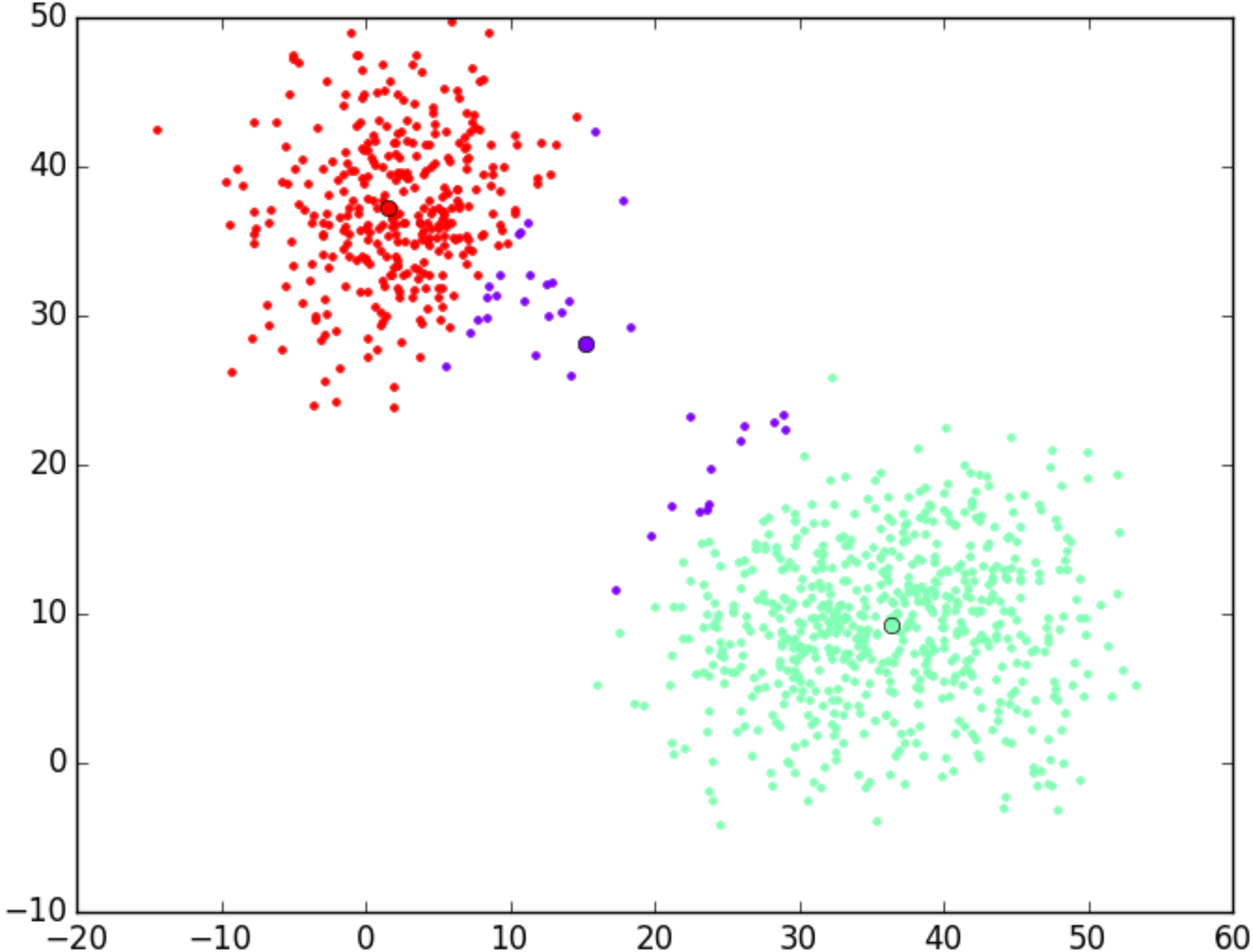
K-means

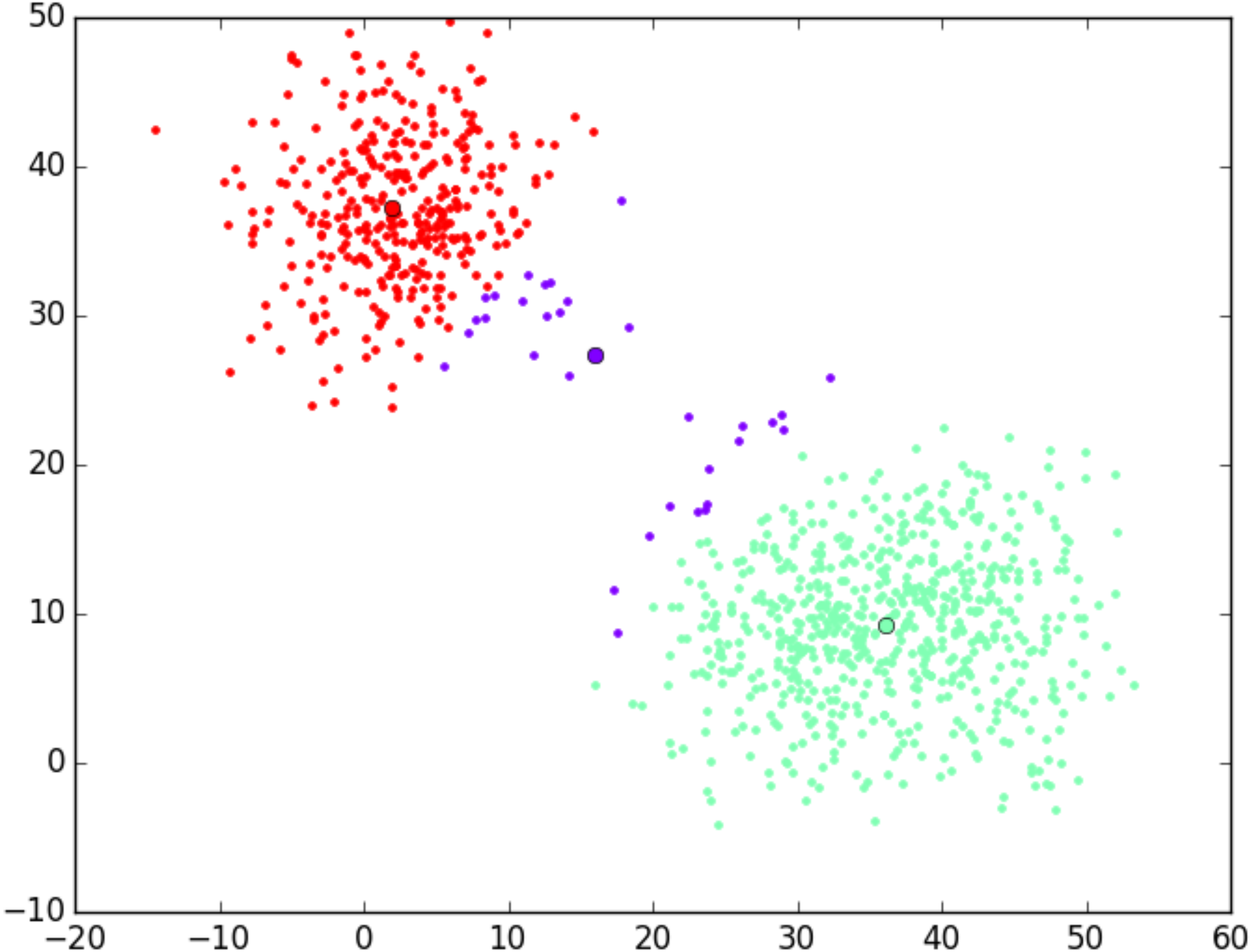
The (standard) algorithm

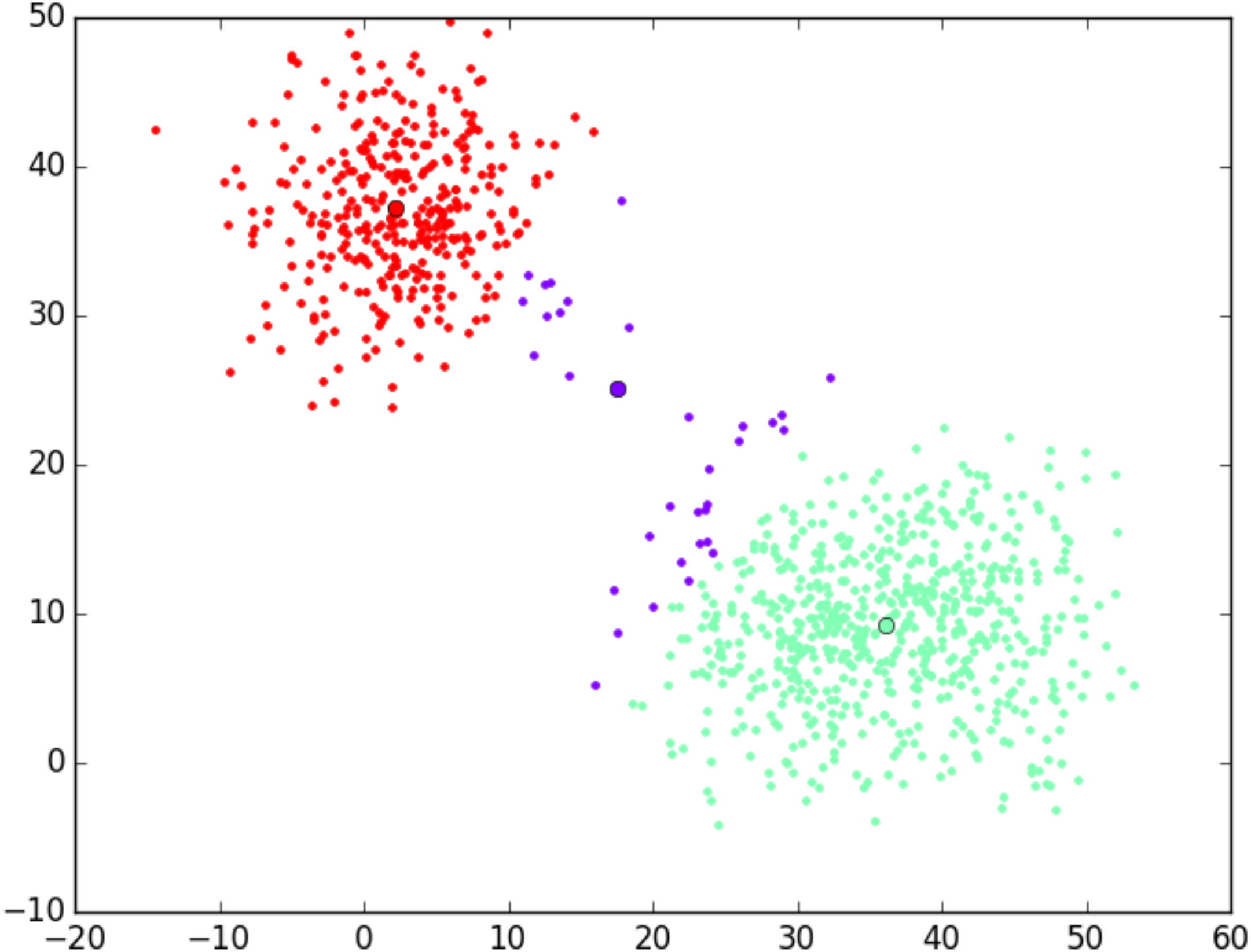
- Puts n data points in d dimensional space into k clusters,
- needs a metric d(-,-).

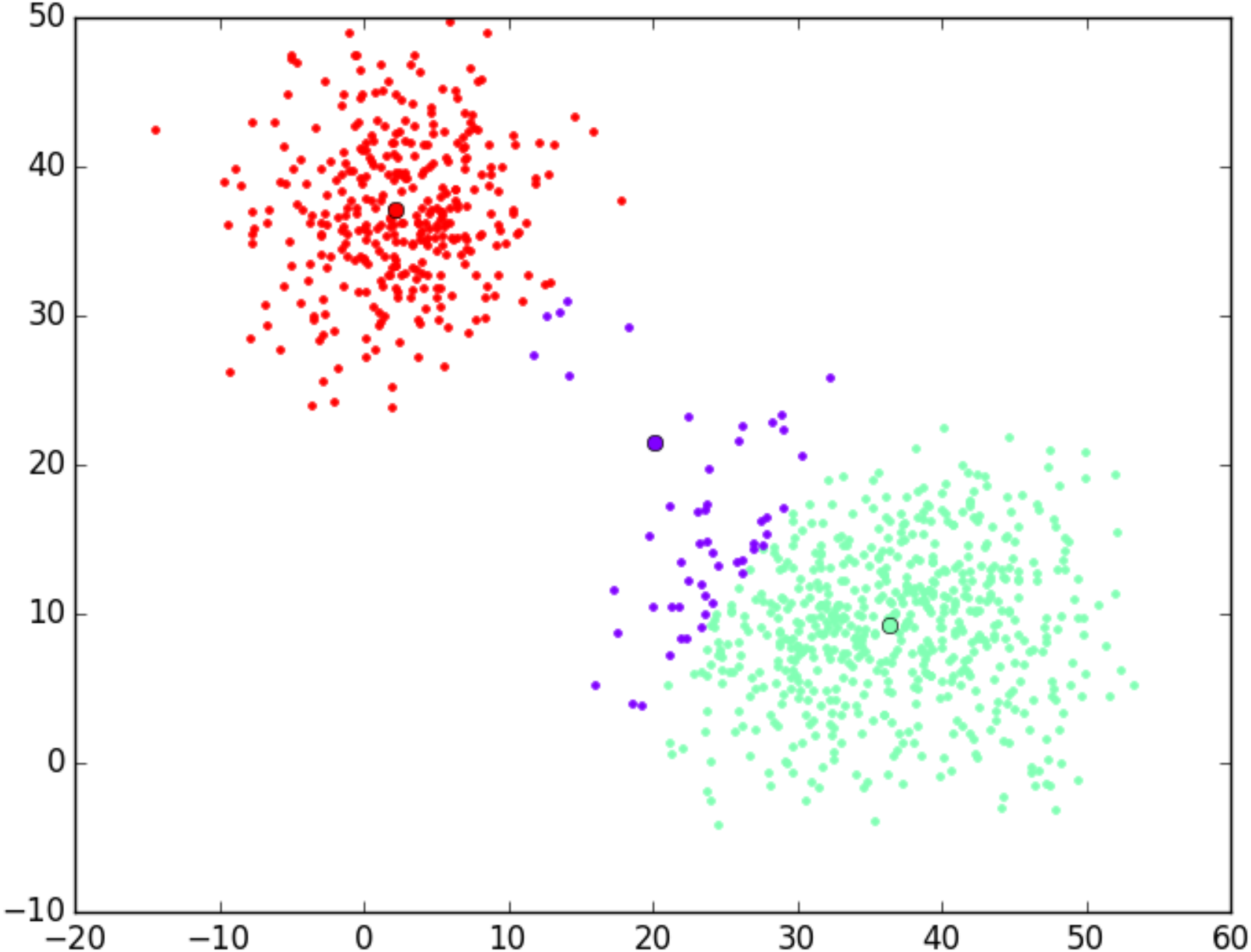
Initialization	Set k initial clusters. At the moment randomly.
Assignment	Assign data points to initial clusters by a responsibility indicator: $r_k^{(n)} = \begin{cases} 1 & \text{, if } \operatorname{argmin}_k\{d(m^{(k)}, x^{(n)}\} = k, \\ 0 & \text{, otherwise.} \end{cases}$
Update	Adjust cluster centers: $m^{(k)} = \frac{\sum_n r_k^{(n)} x^{(n)}}{\sum_n r_k^{(n)}}$
Go to assignment step	… until stopping criterion is reached. Usually, when the changes in the assignment step are reasonably small.

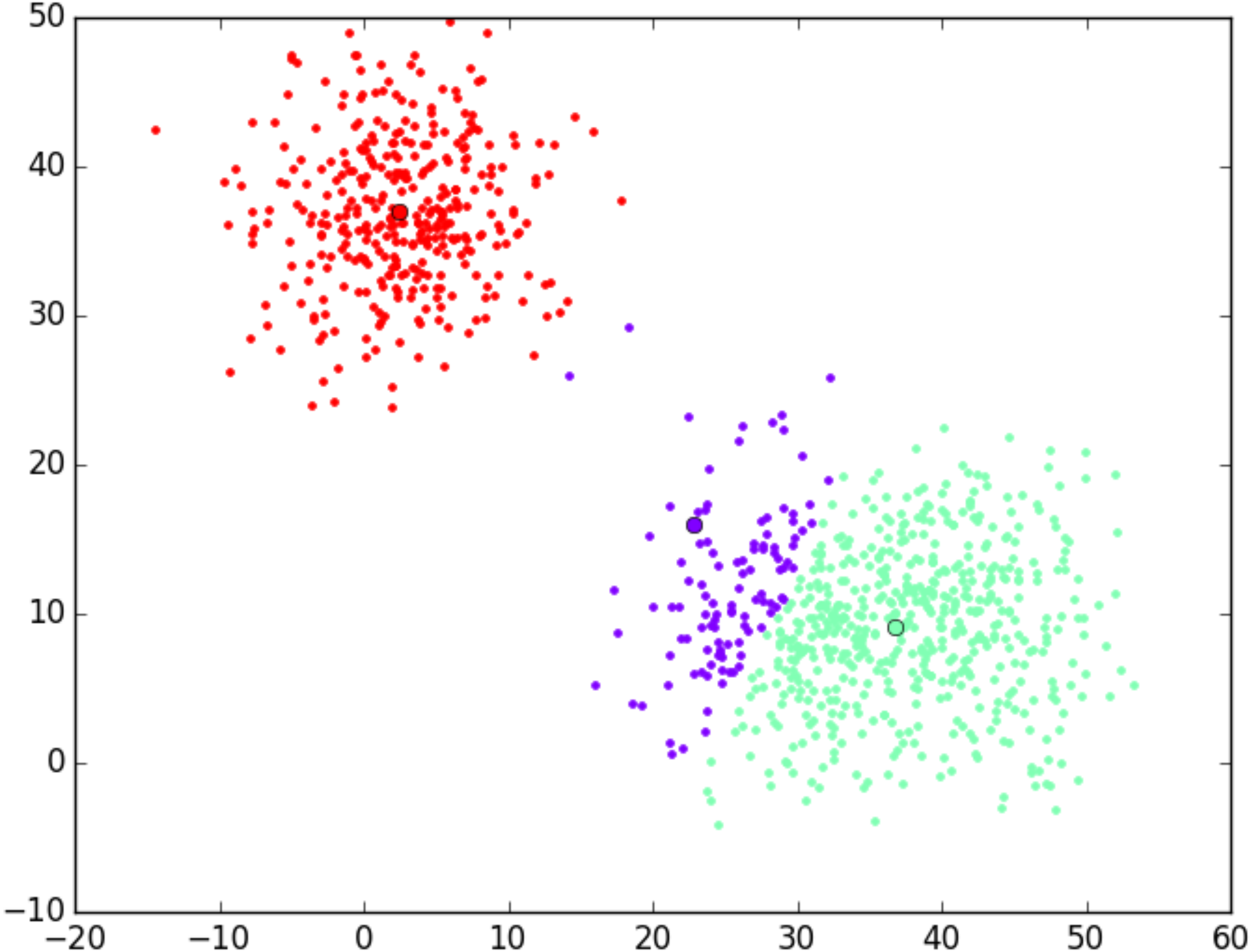

50
40
30
20
10
0
−10
−20
−10
0
10
20
30
40
50
60

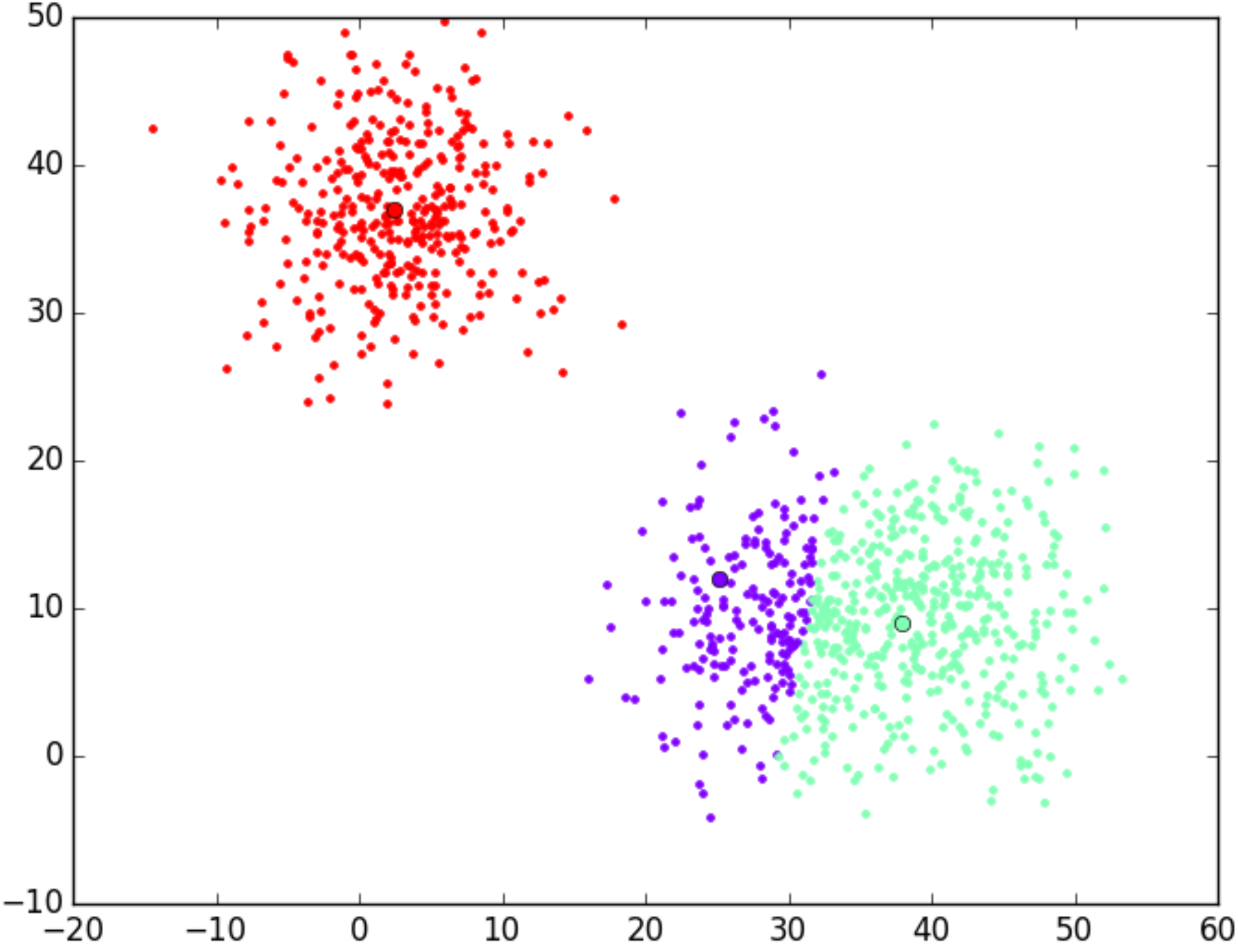

50
40
30
20
10
0
−10
−20
−10
0
10
20
30
40
50
60

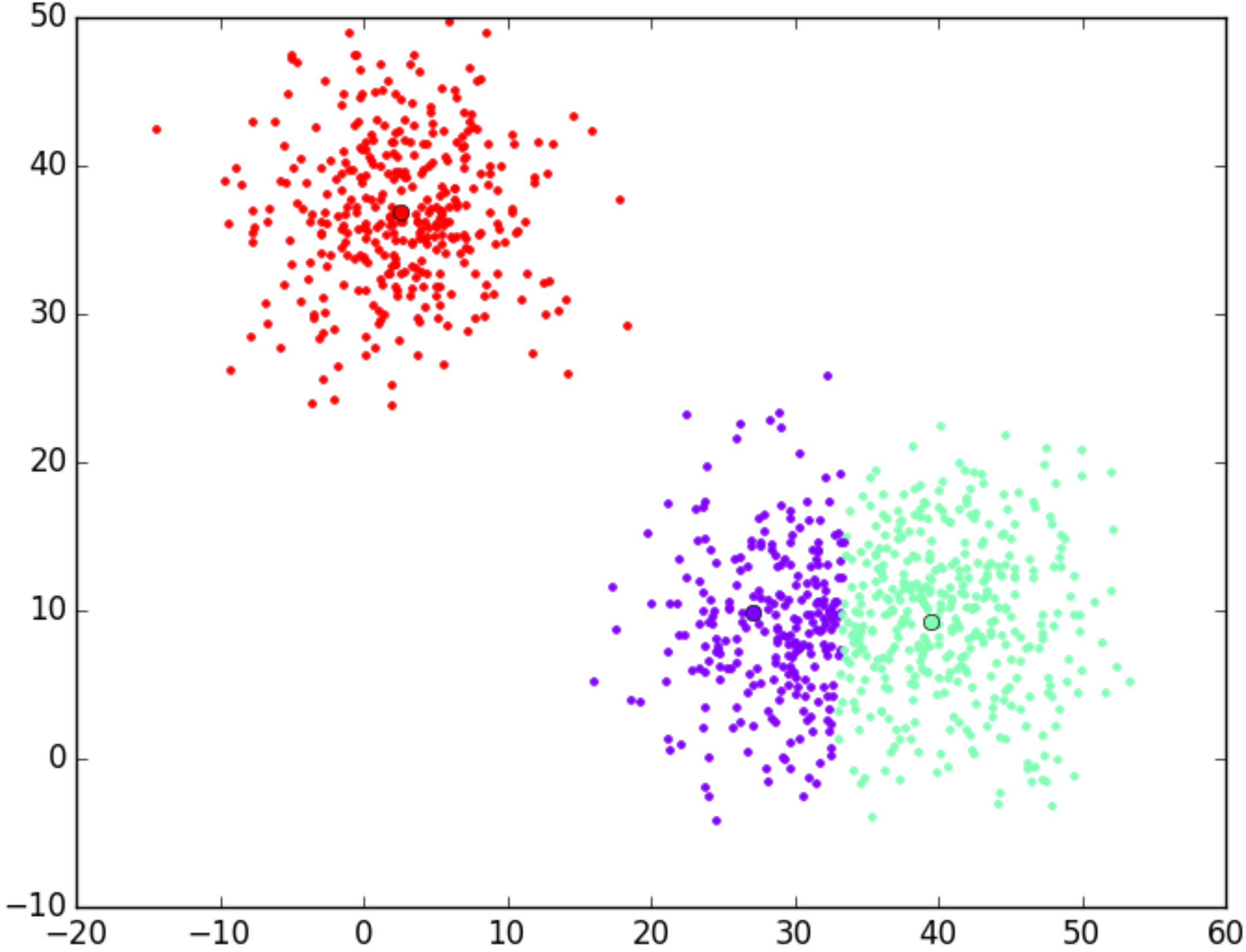

50
40
30
20
10
0
−10
−20
−10
0
10
20
30
40
50
60

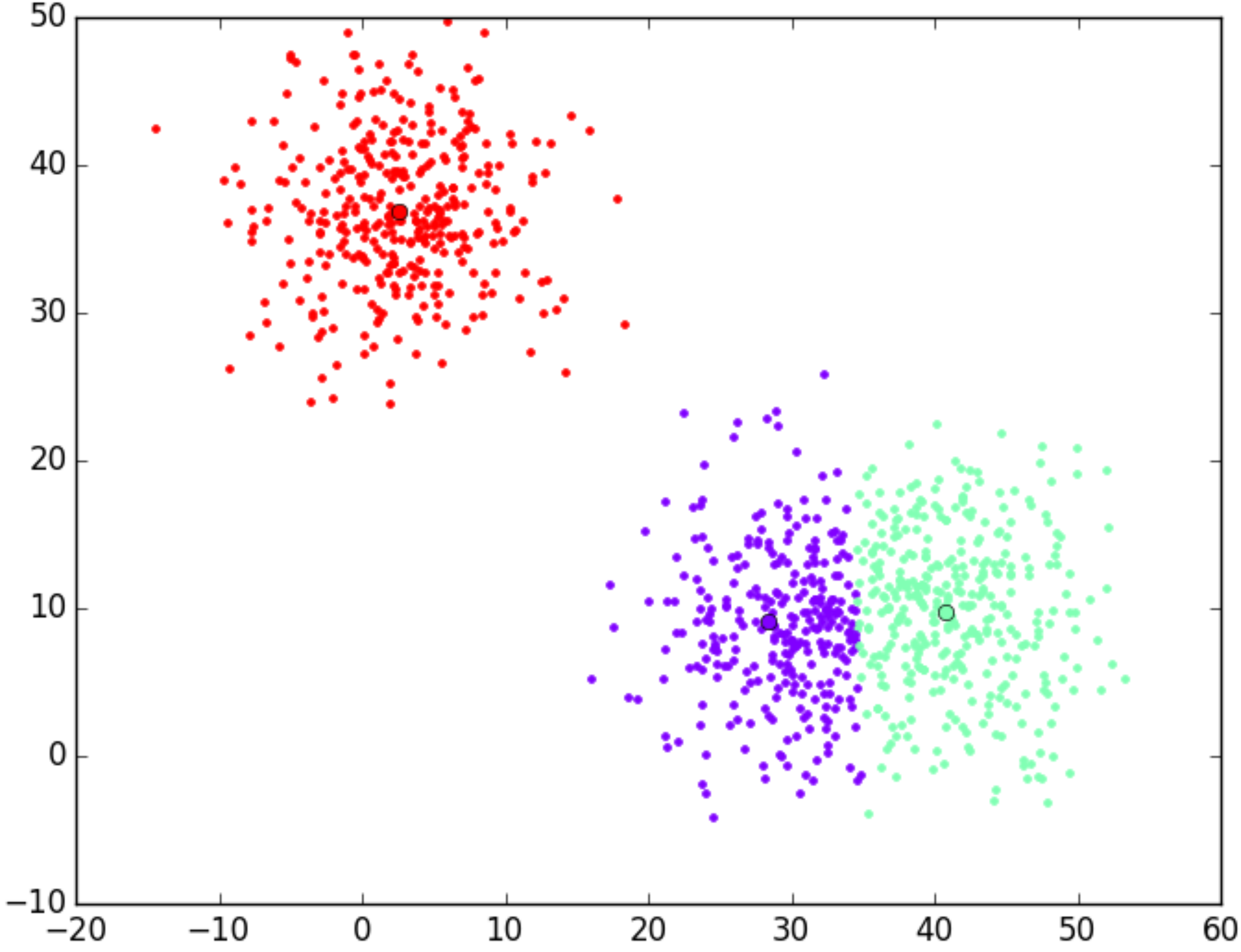

50
40
30
20
10
0
−10
−20
−10
0
10
20
30
40
50
60

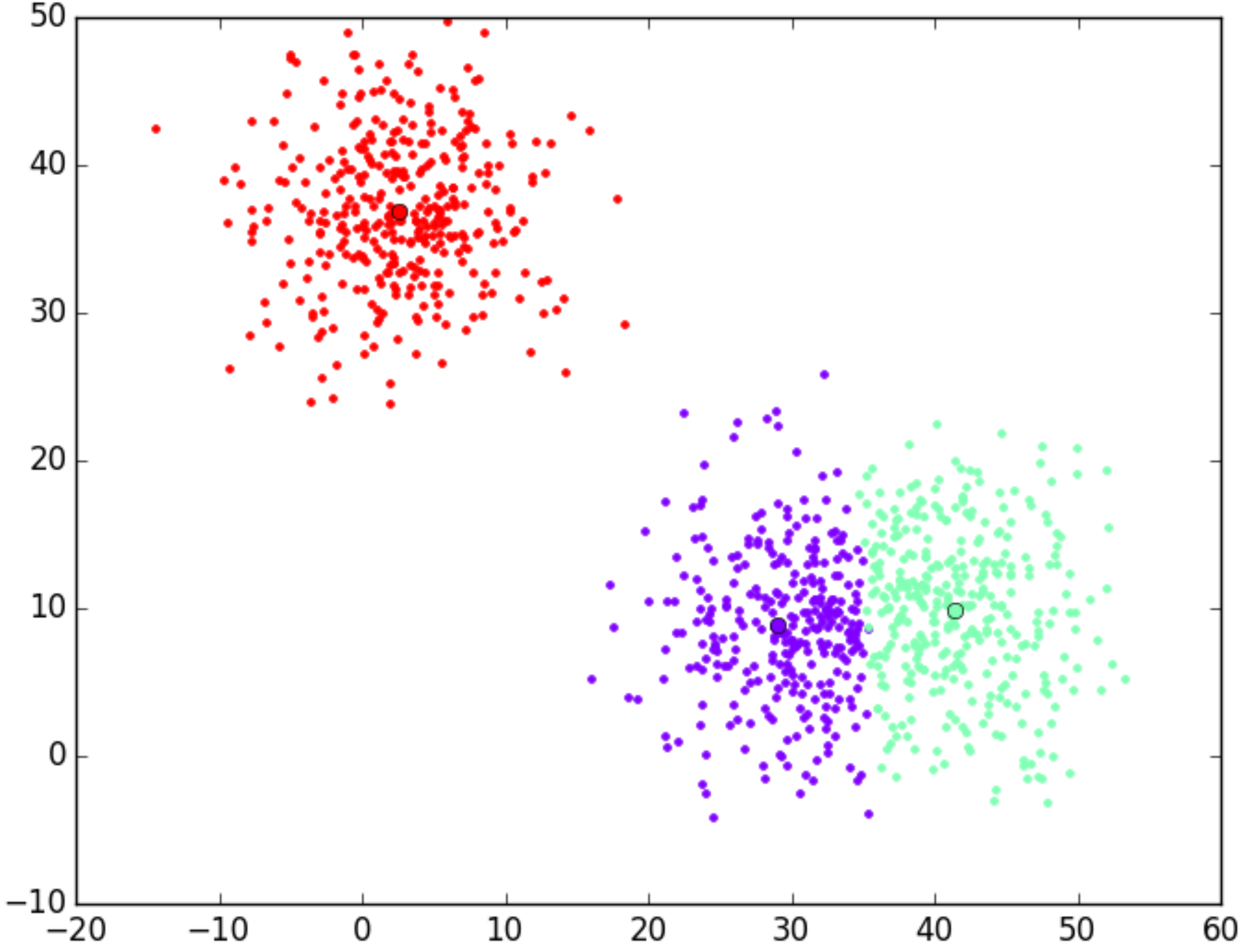

50
40
30
20
10
0
−10
−20
−10
0
10
20
30
40
50
60

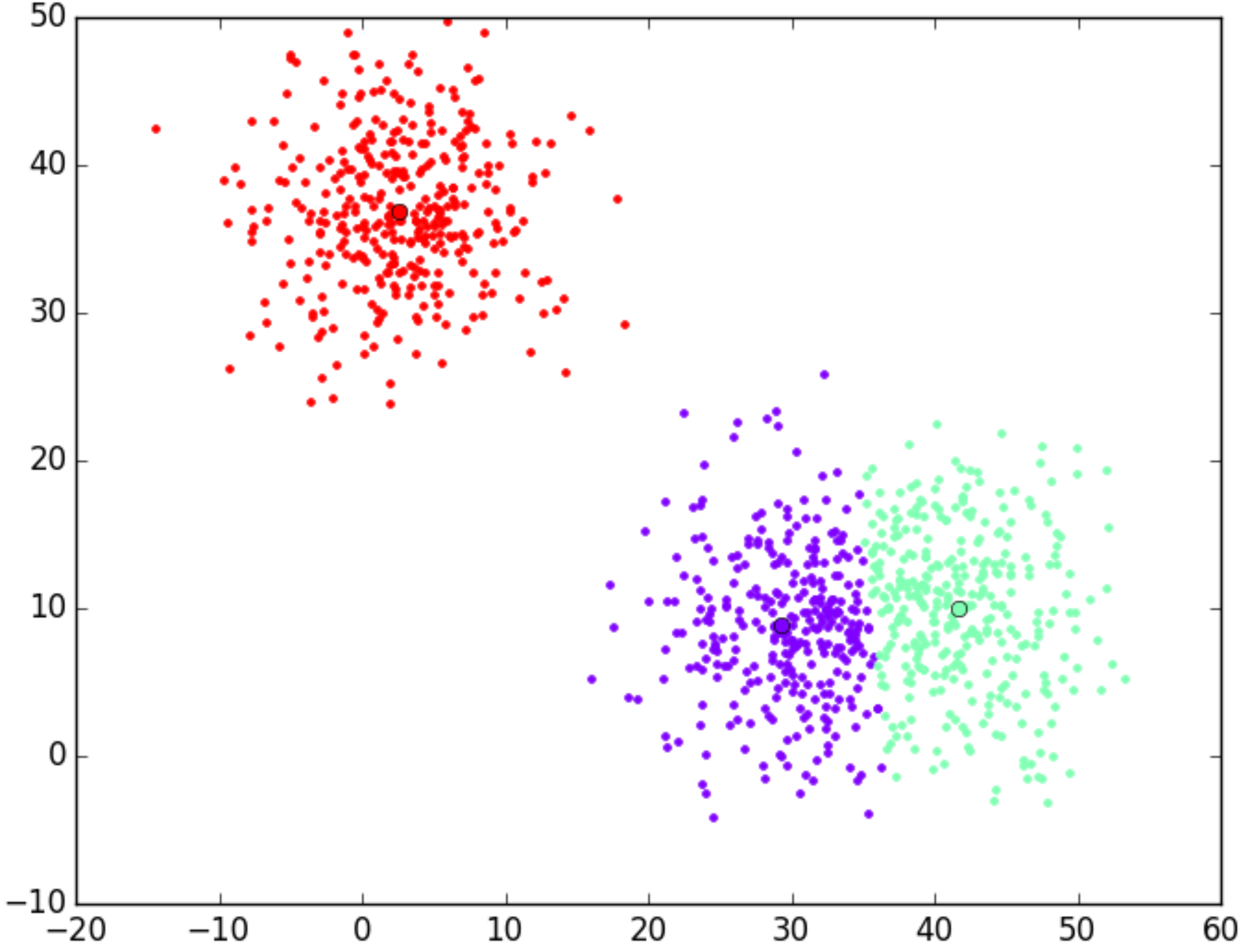

50
40
30
20
10
0
−10
−20
−10
0
10
20
30
40
50
60

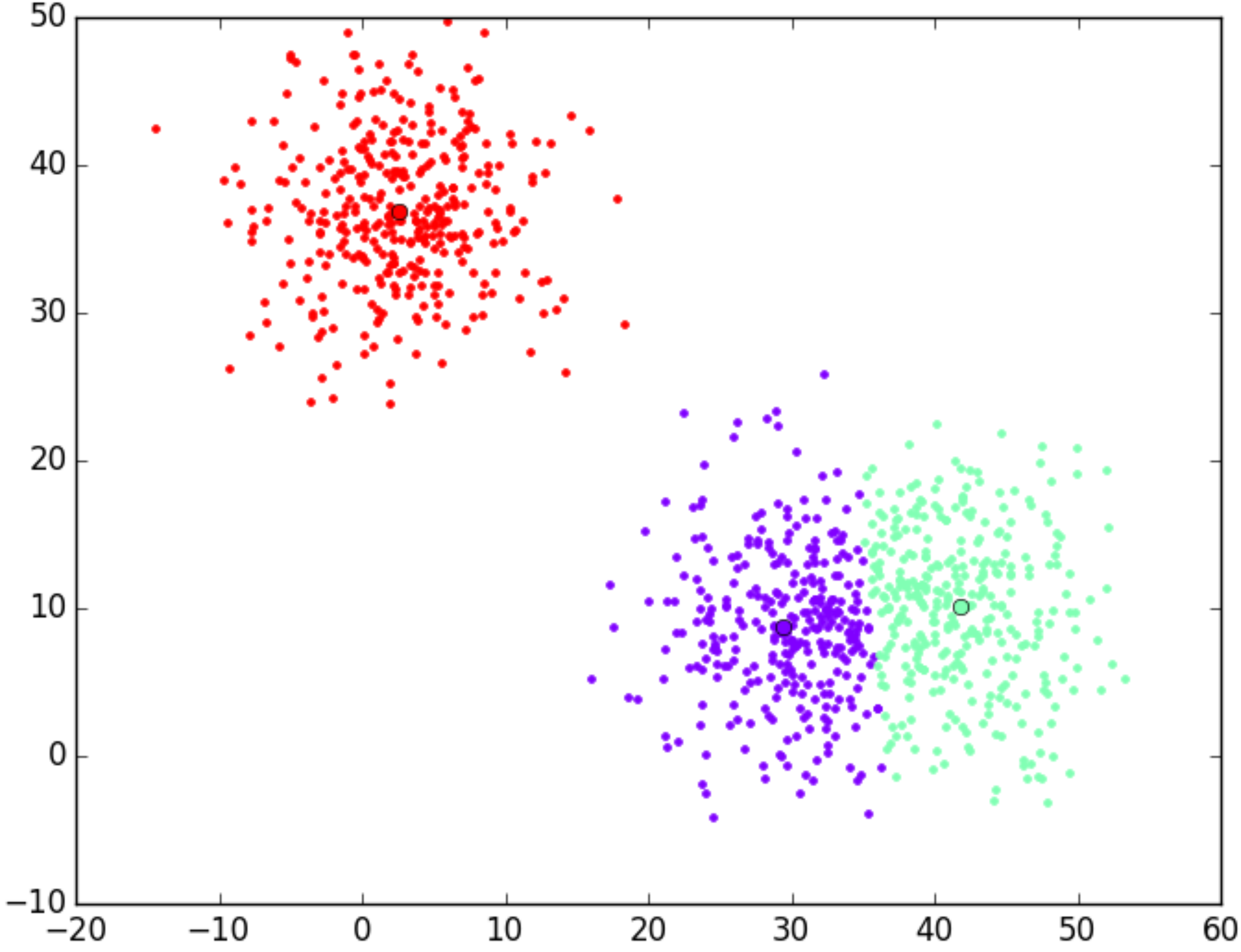

50
40
30
20
10
0
−10
−20
−10
0
10
20
30
40
50
60


50
40
30
20
10
0
−10
−20
−10
0
10
20
30
40
50
60


50
40
30
20
10
0
−10
−20
−10
0
10
20
30
40
50
60


50
40
30
20
10
0
−10
−20
−10
0
10
20
30
40
50
60


50
40
30
20
10
0
−10
−20
−10
0
10
20
30
40
50
60


50
40
30
20
10
0
−10
−20
−10
0
10
20
30
40
50
60

Problems

- Number of cluster centers
- Placement of the initial centers
- Stiffness of the assignment step
- Size of the data
- Performance

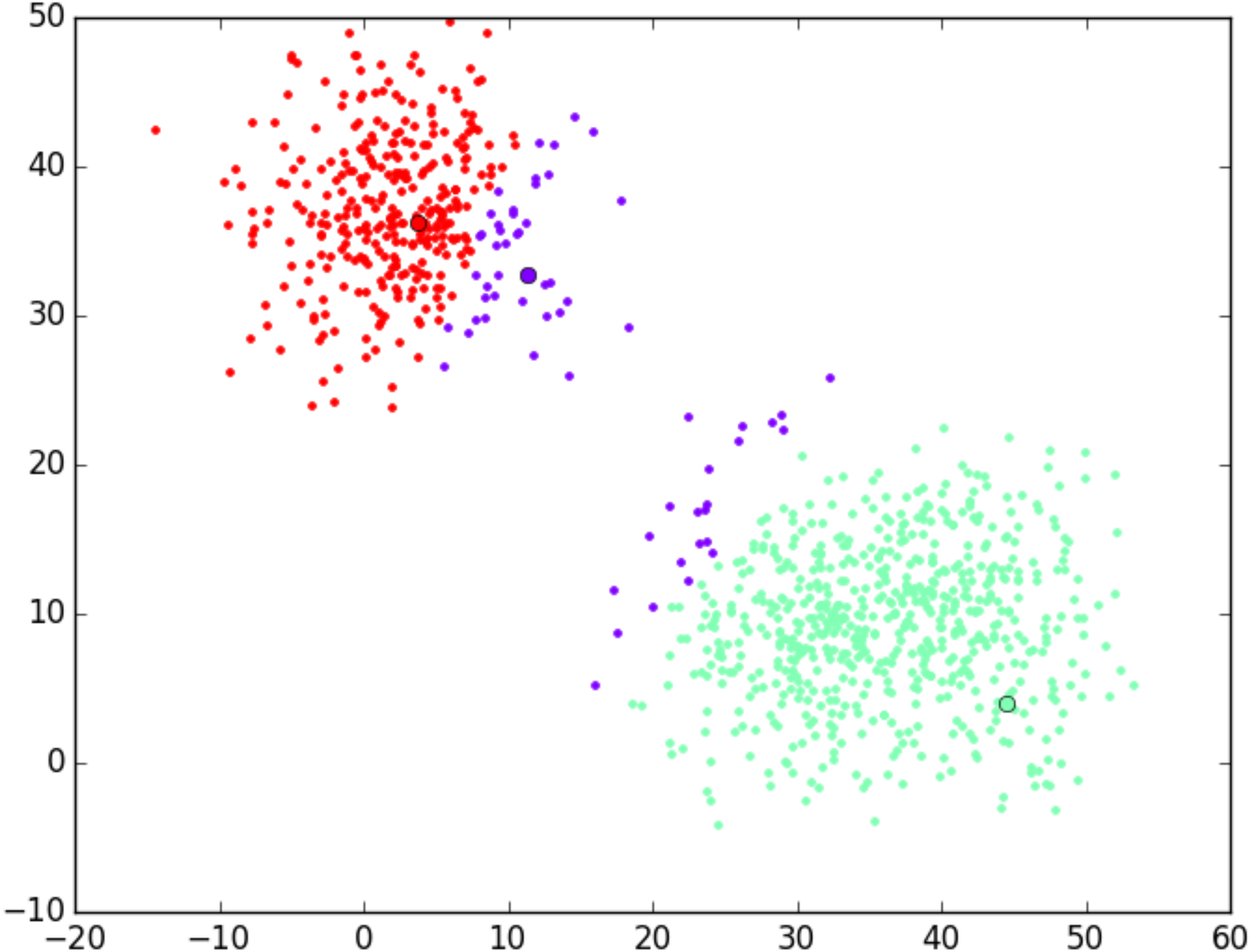
Stiffness problem / soft k-means

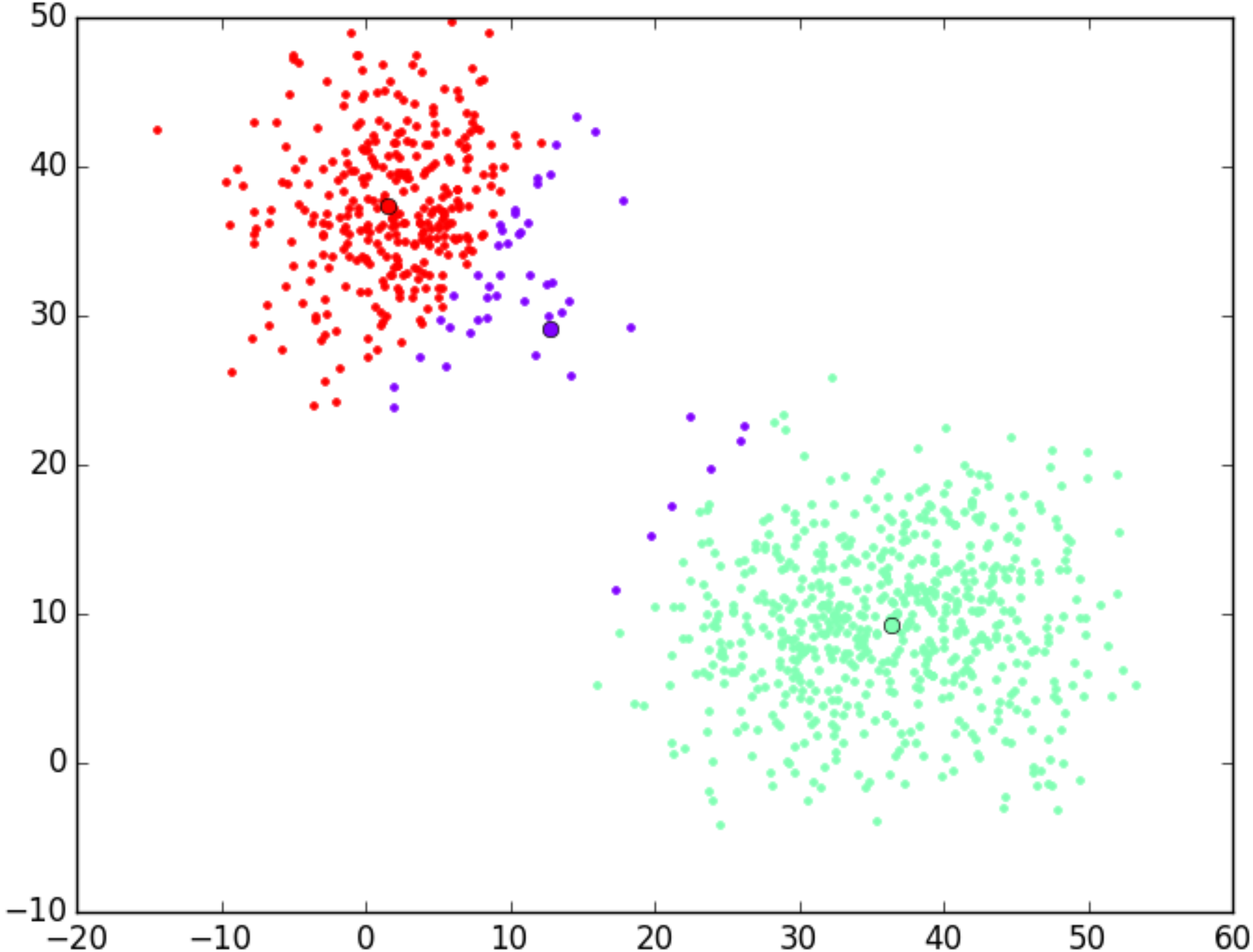
Replace 0-1-responsibility-function by a weighted sum:

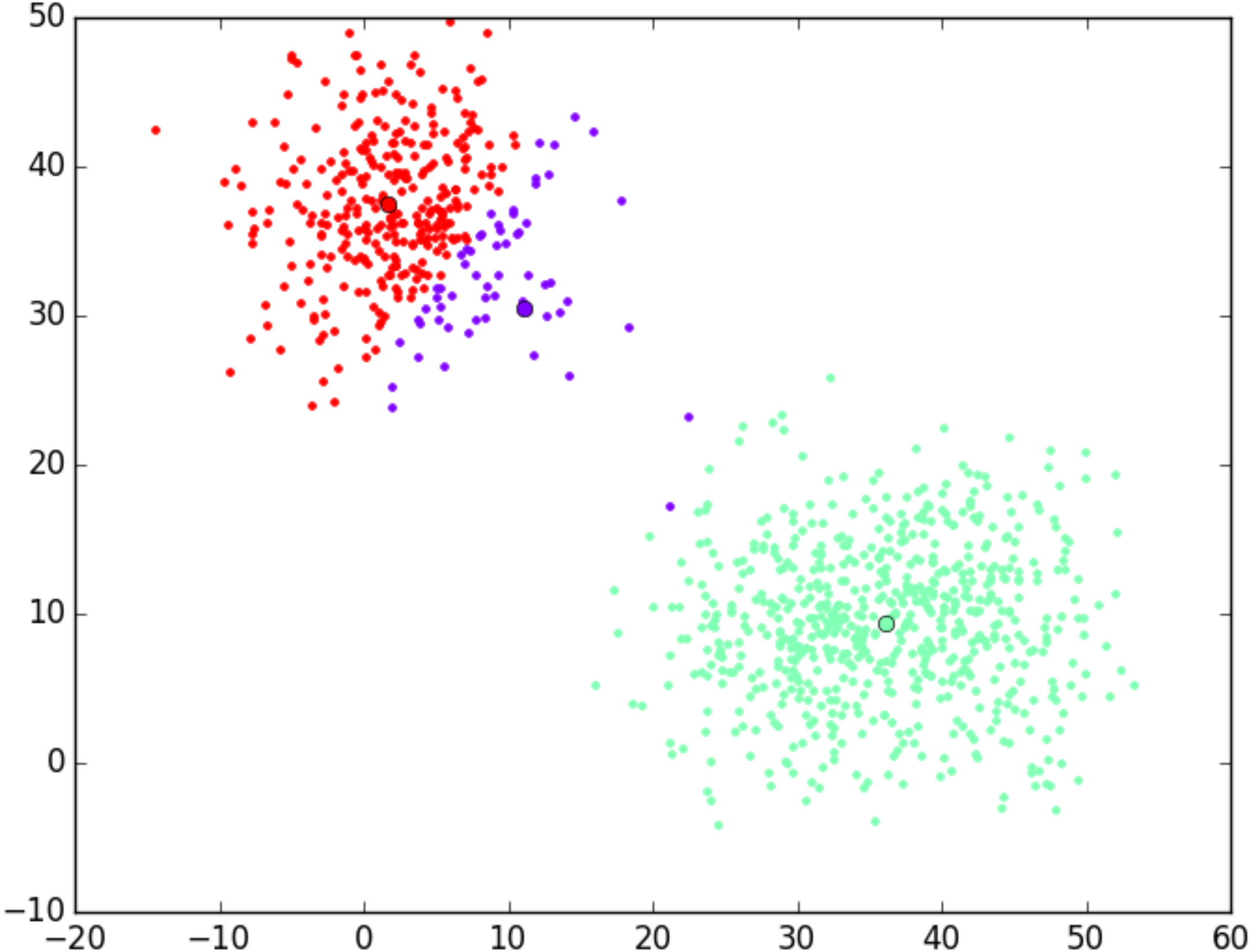
$$r_k^{(n)} = \frac{\exp(-\beta d(m^{(k)}, x^{(n)}))}{\sum_{l=1}^{k} \exp(-\beta d(m^{(l)}, x^{(n)}))}$$

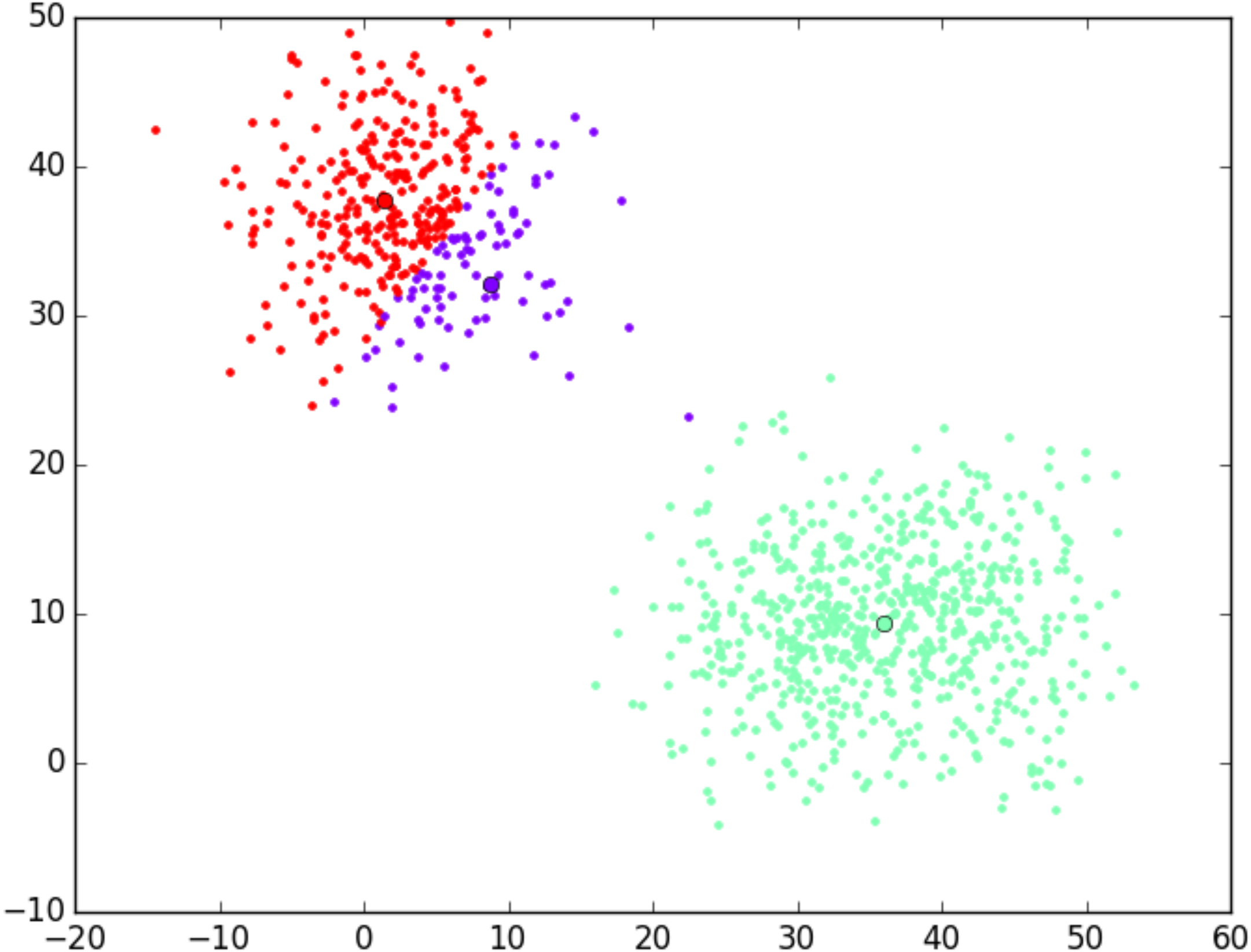
- includes a partial role of points close to the border between two or more clusters
- parameter β is unknown
- gives a percentage value of responsibility, as

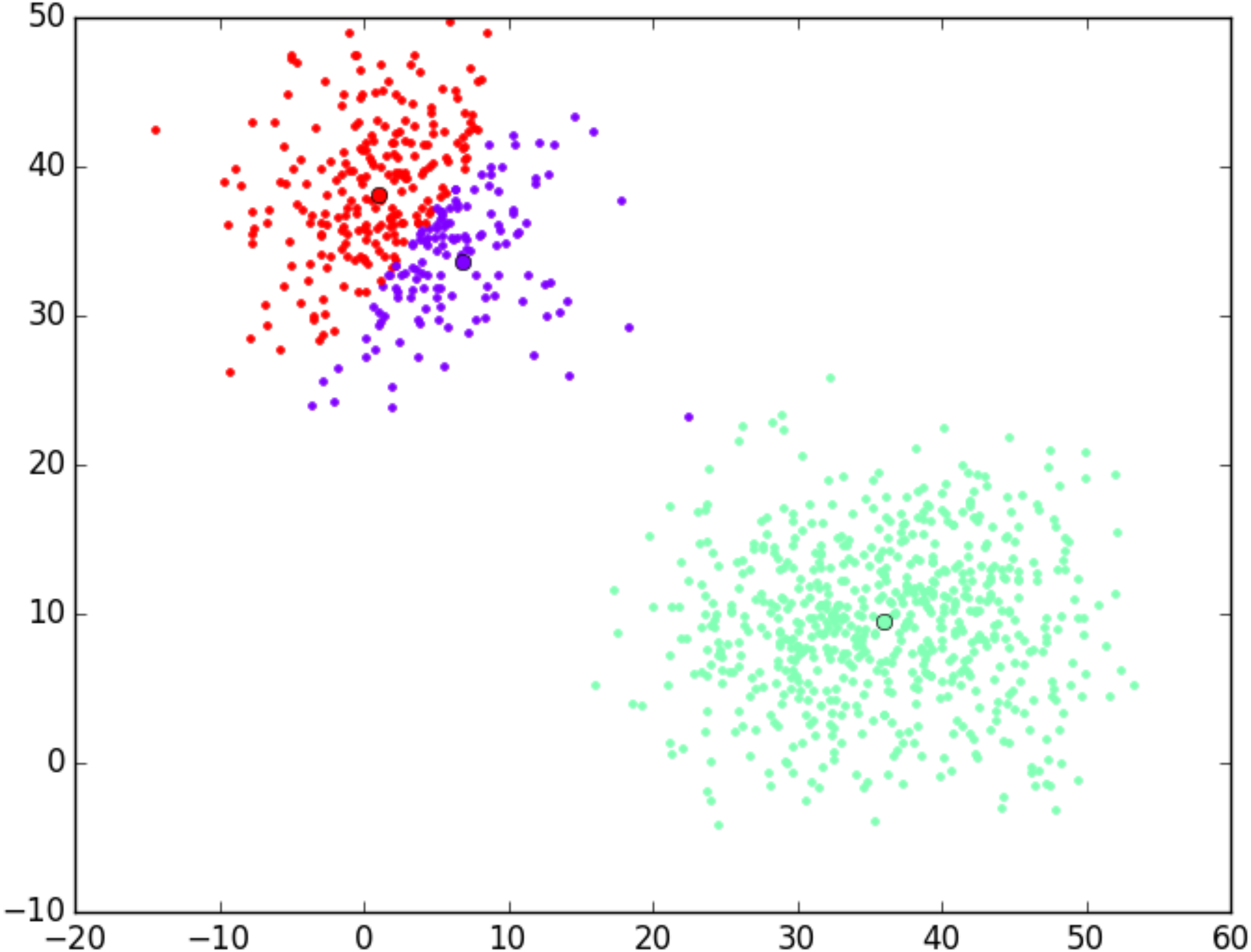
$$\sum_k r_k^{(n)} = 1$$

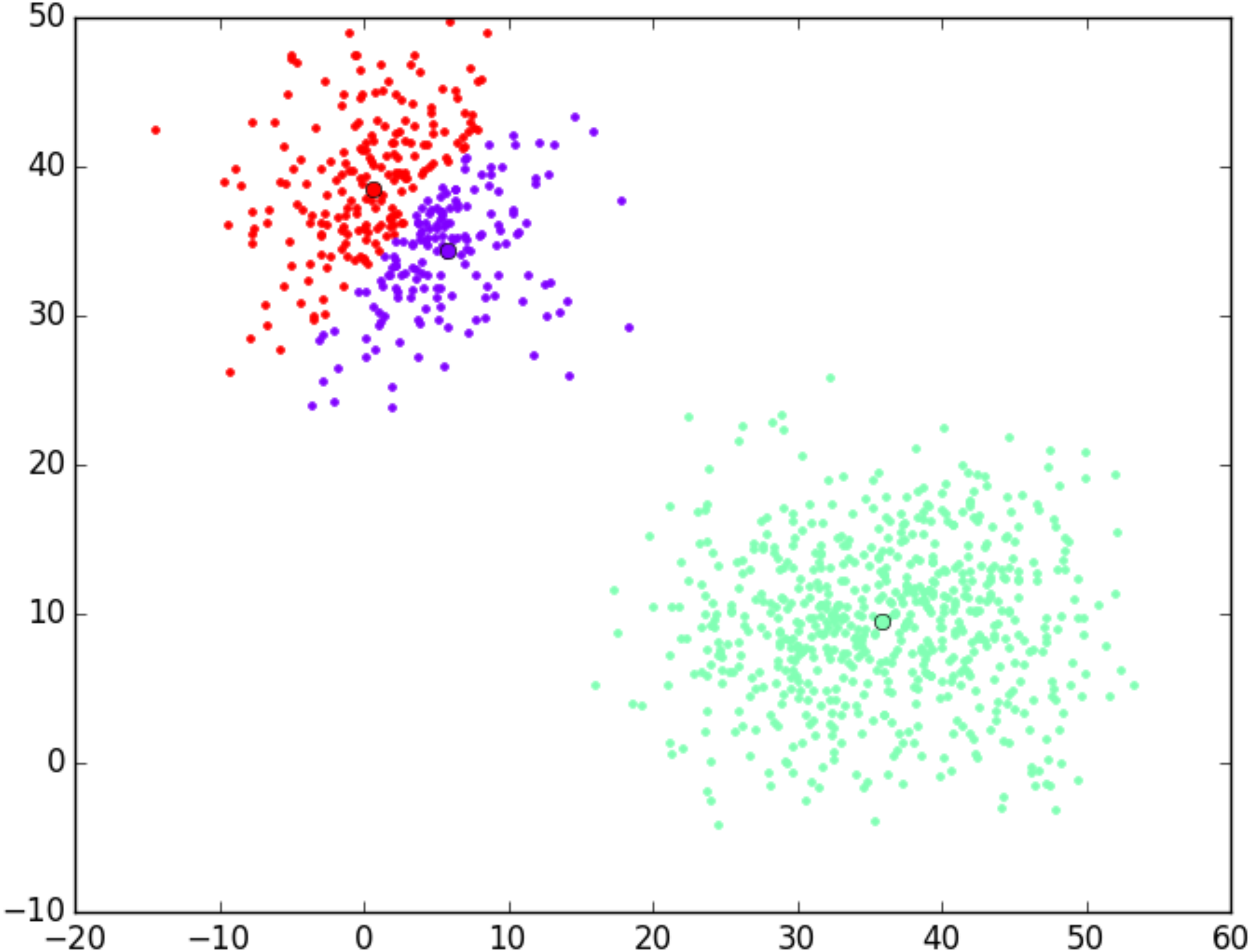

50
40
30
20
10
0
−10
−20
−10
0
10
20
30
40
50
60

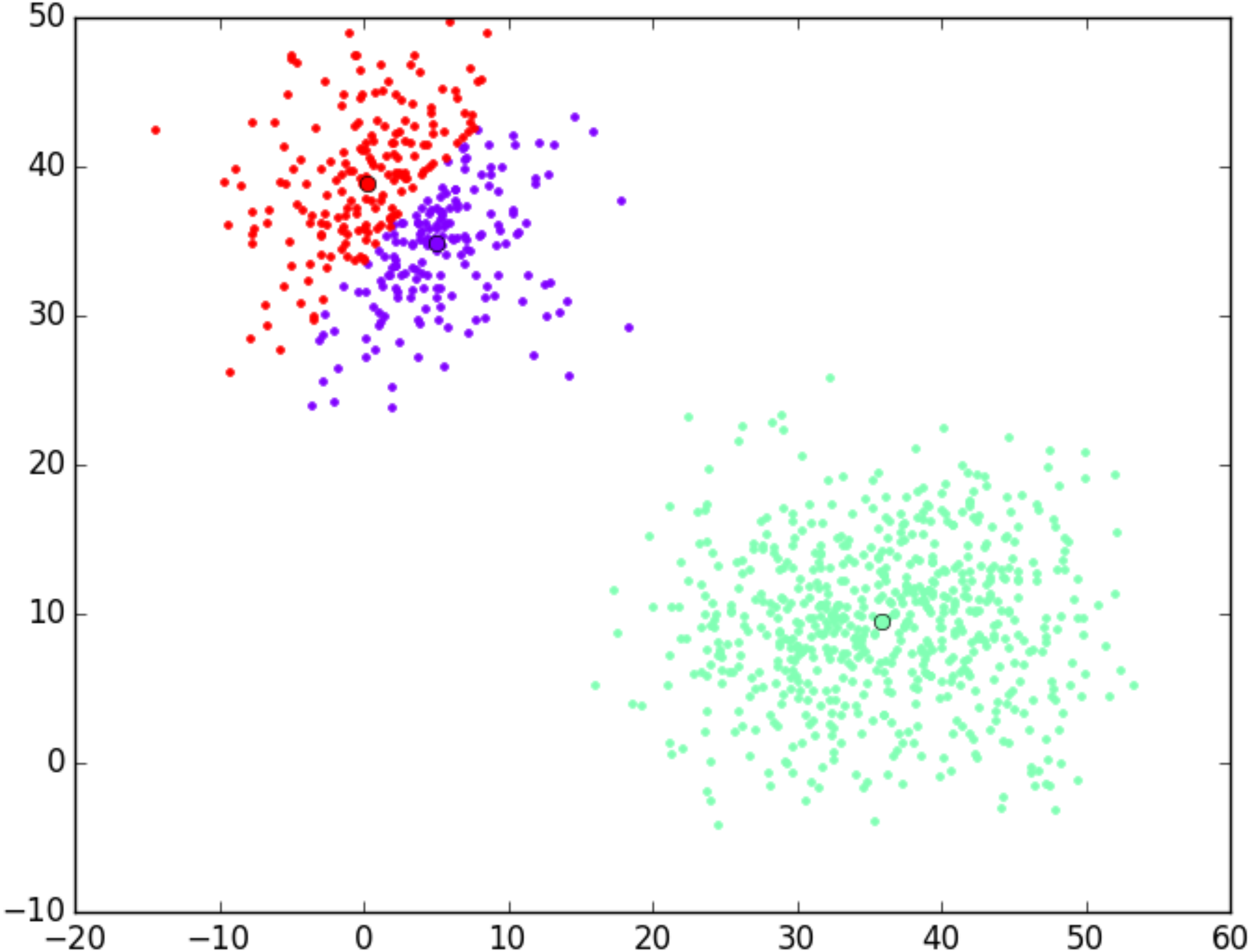

50
40
30
20
10
0
−10
−20
−10
0
10
20
30
40
50
60

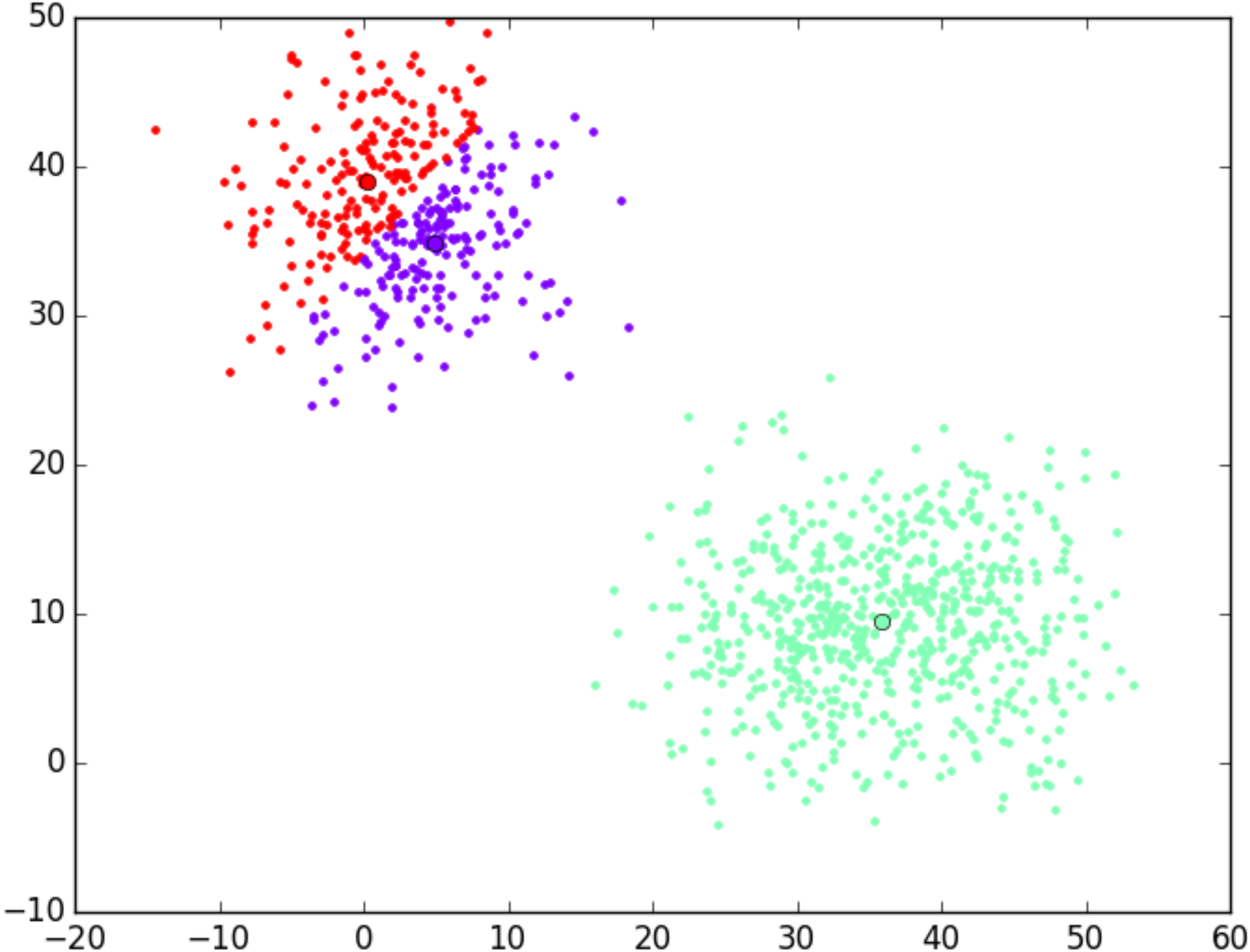

50
40
30
20
10
0
−10
−20
−10
0
10
20
30
40
50
60

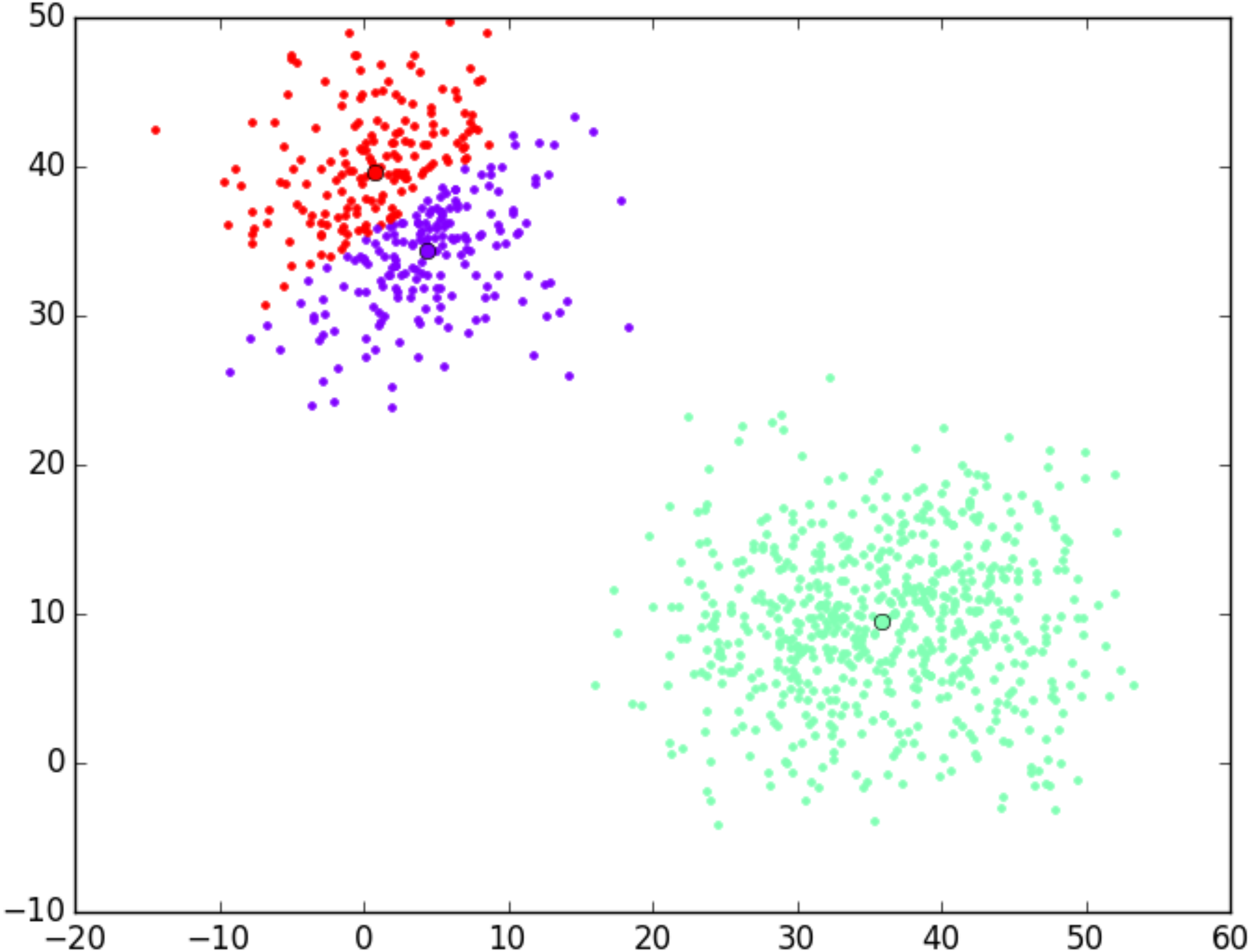

50
40
30
20
10
0
−10
−20
−10
0
10
20
30
40
50
60


50
40
30
20
10
0
−10
−20
−10
0
10
20
30
40
50
60


50
40
30
20
10
0
−10
−20
−10
0
10
20
30
40
50
60

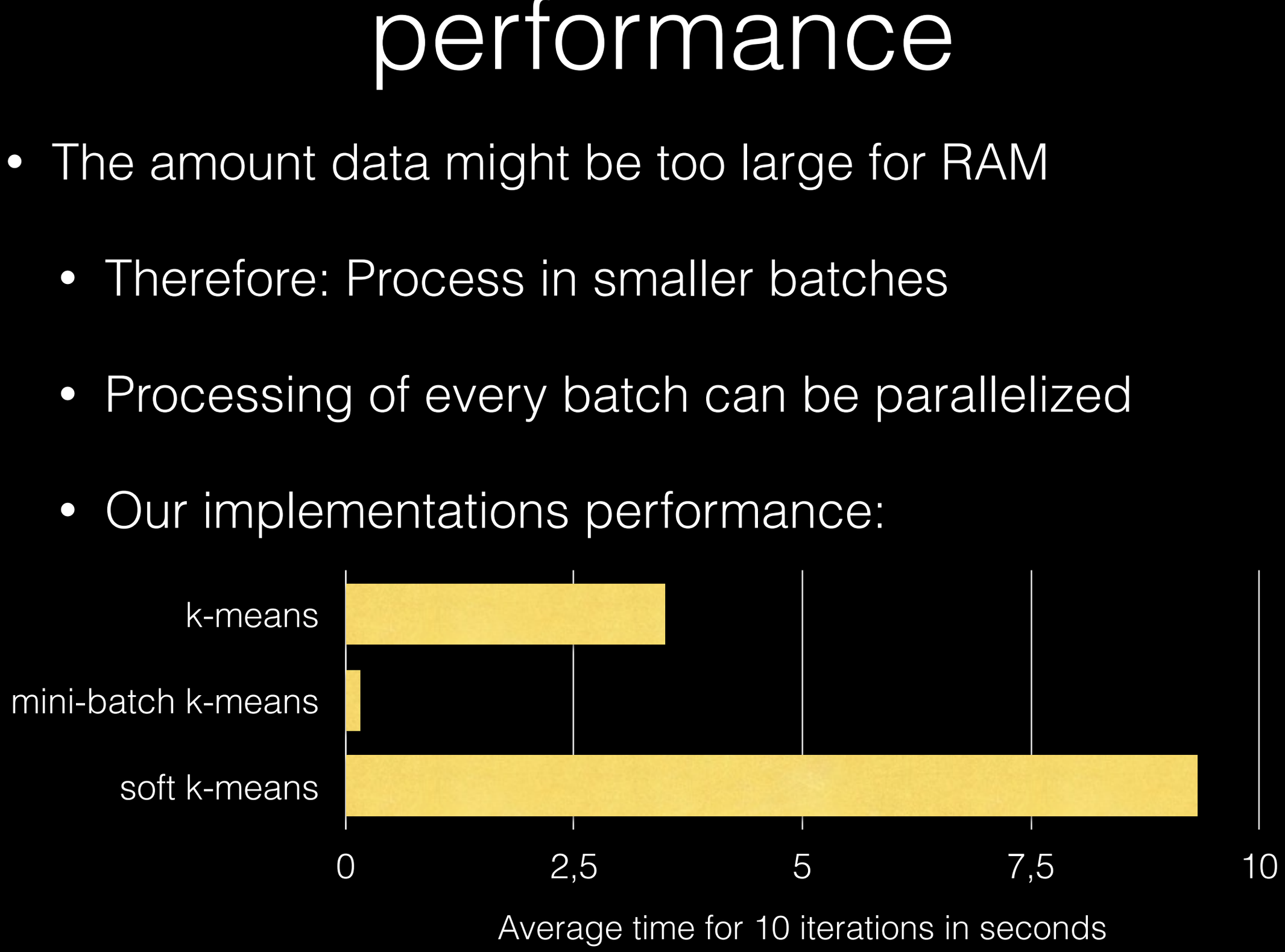

50
40
30
20
10
0
−10
−20
−10
0
10
20
30
40
50
60


50
40
30
20
10
0
−10
−20
−10
0
10
20
30
40
50
60


50
40
30
20
10
0
−10
−20
−10
0
10
20
30
40
50
60

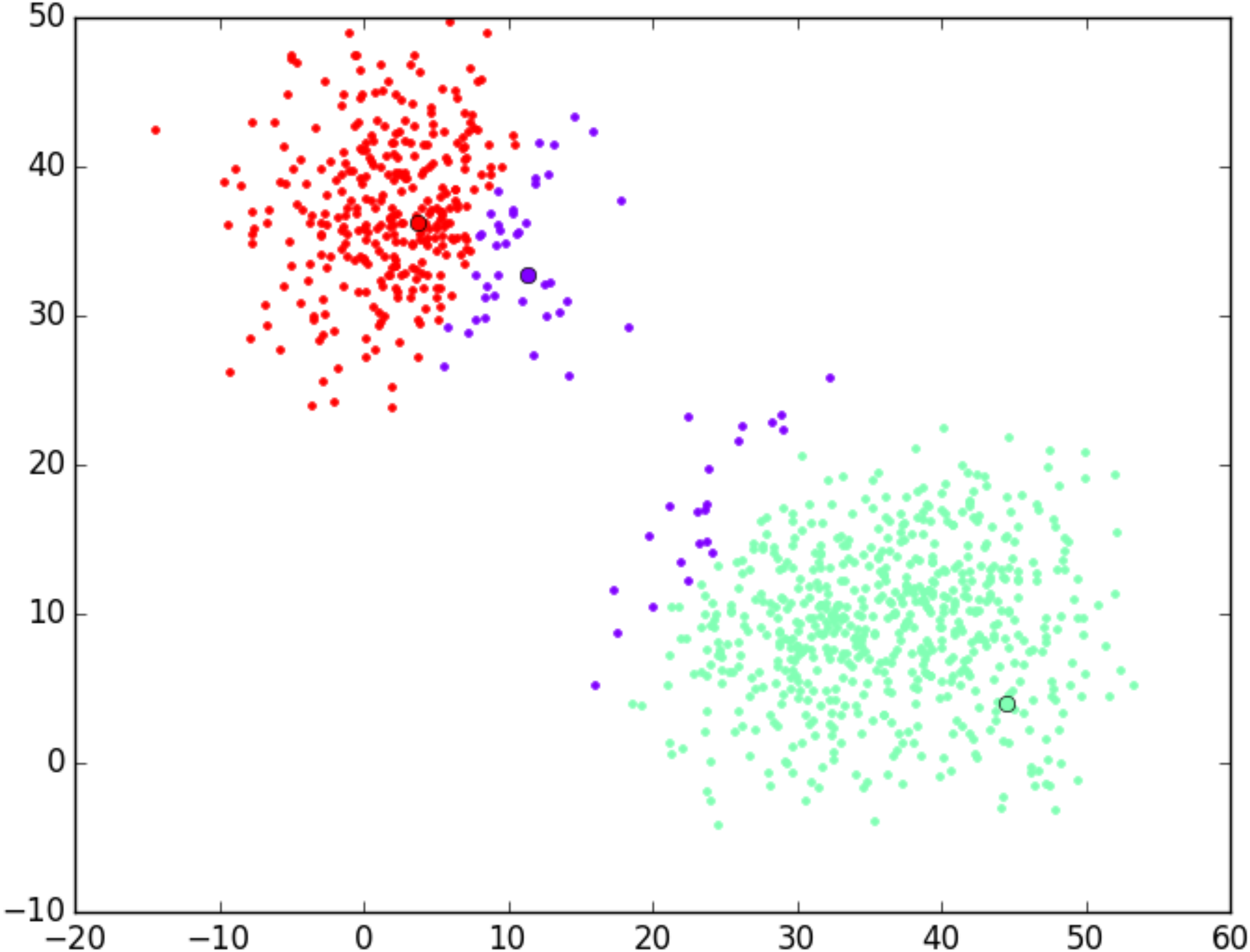
Size of the data / performance

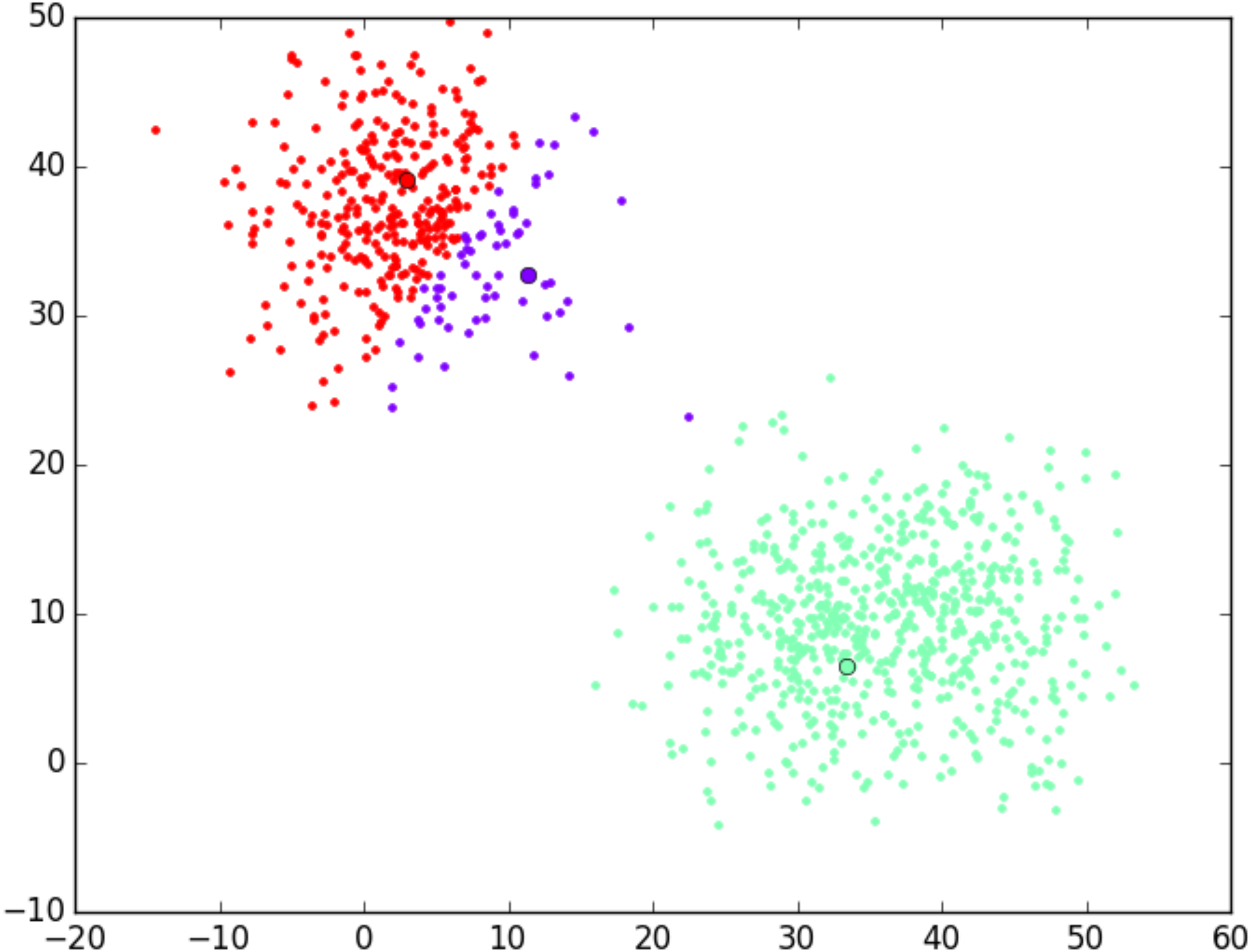
- The amount data might be too large for RAM
  - Therefore: Process in smaller batches
  - Processing of every batch can be parallelized
  - Our implementations performance:

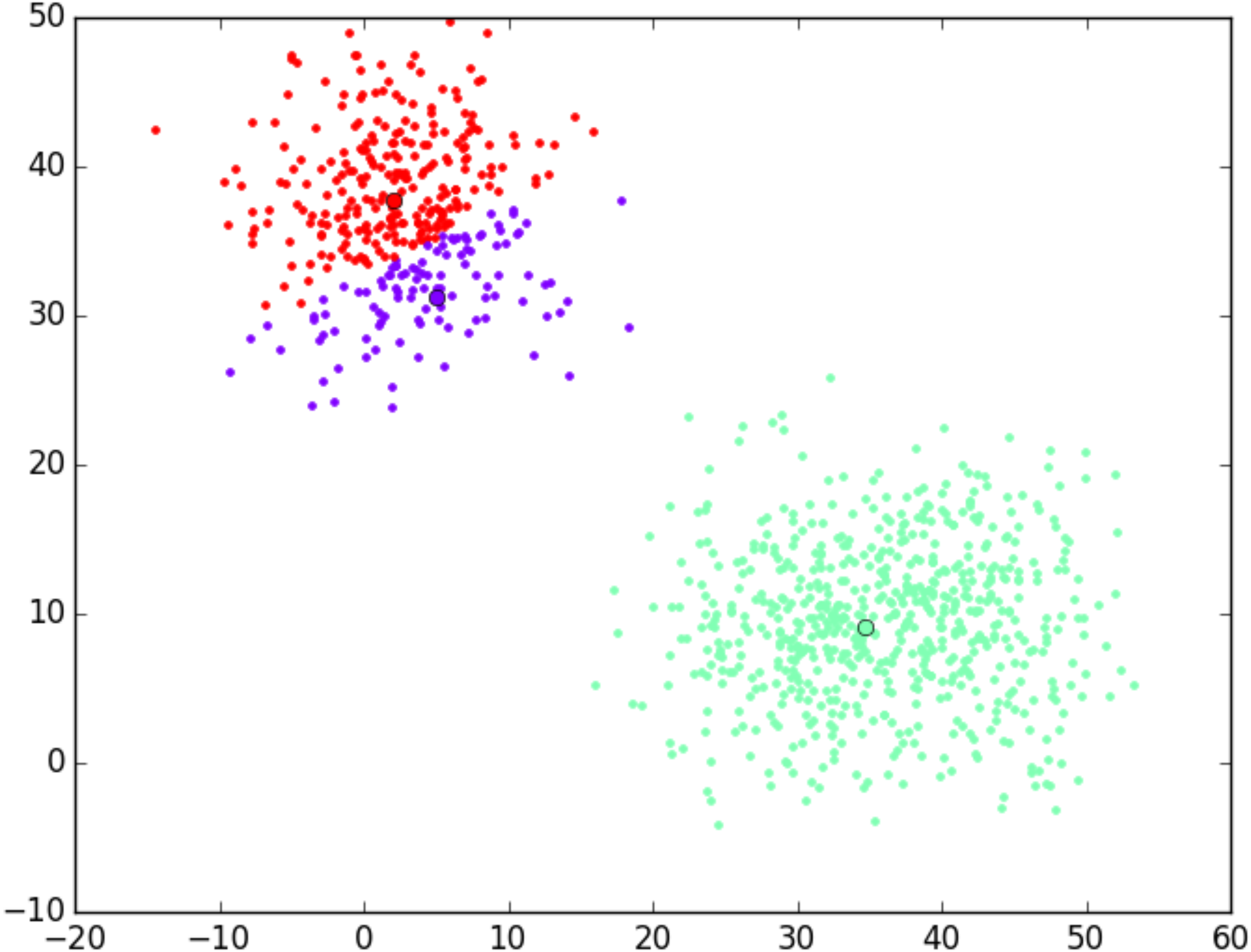


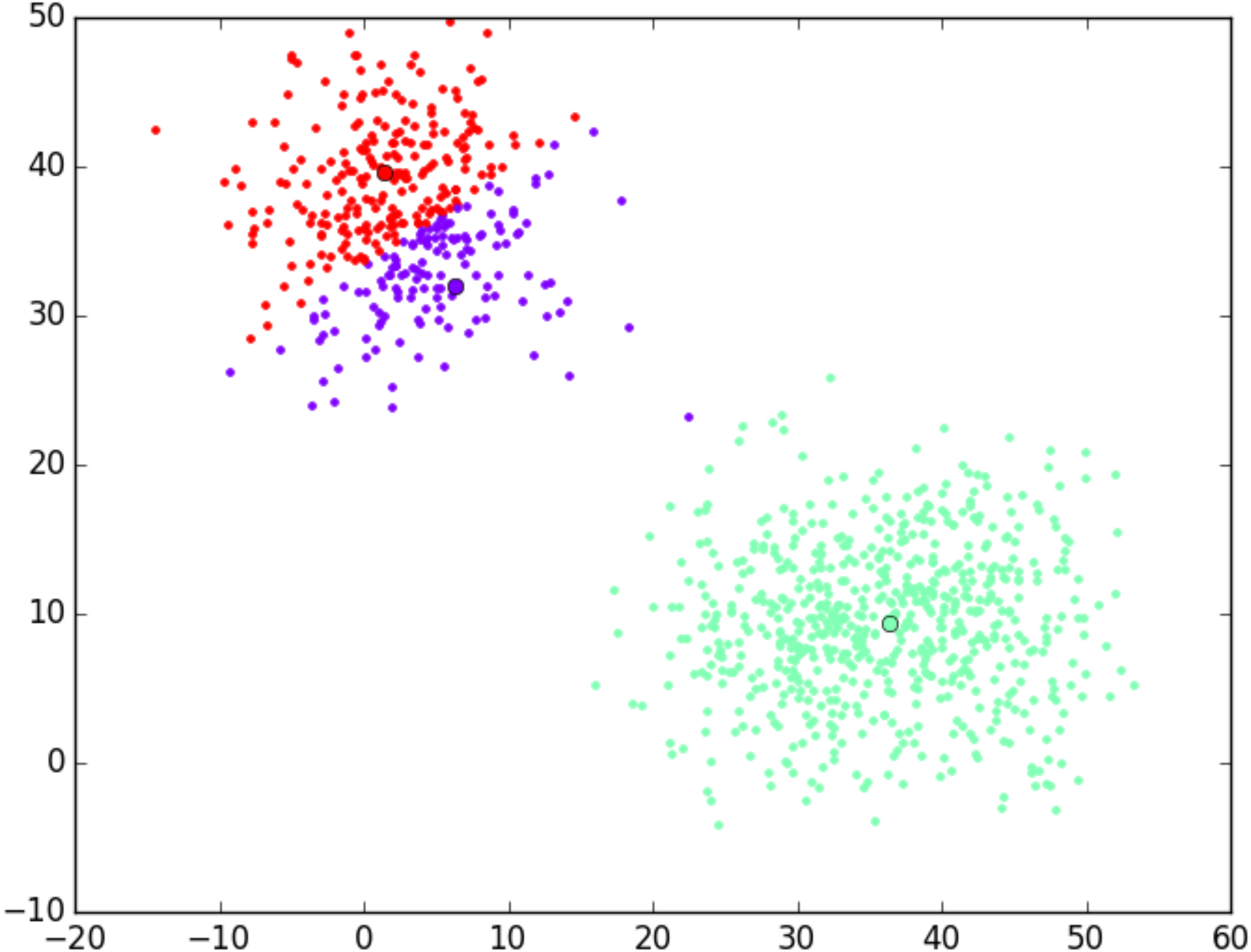
Mini-Batch k-means

- Uses a randomly selected subset of the data
- Initializes the cluster centers by a random selection of the data points
- Might lead to oscillations if the selected subset is too small


50
40
30
20
10
0
−10
−20
−10
0
10
20
30
40
50
60


50
40
30
20
10
0
−10
−20
−10
0
10
20
30
40
50
60


50
40
30
20
10
0
−10
−20
−10
0
10
20
30
40
50
60


50
40
30
20
10
0
−10
−20
−10
0
10
20
30
40
50
60

Implementation