Lecture 9

Scalable K-means

Haiping Lu

http://www.dcs.shef.ac.uk/~haiping

COM6012 Scalable Machine Learning

Spring 2018

Week 9 Contents

- **Introduction to Cluster Analysis**

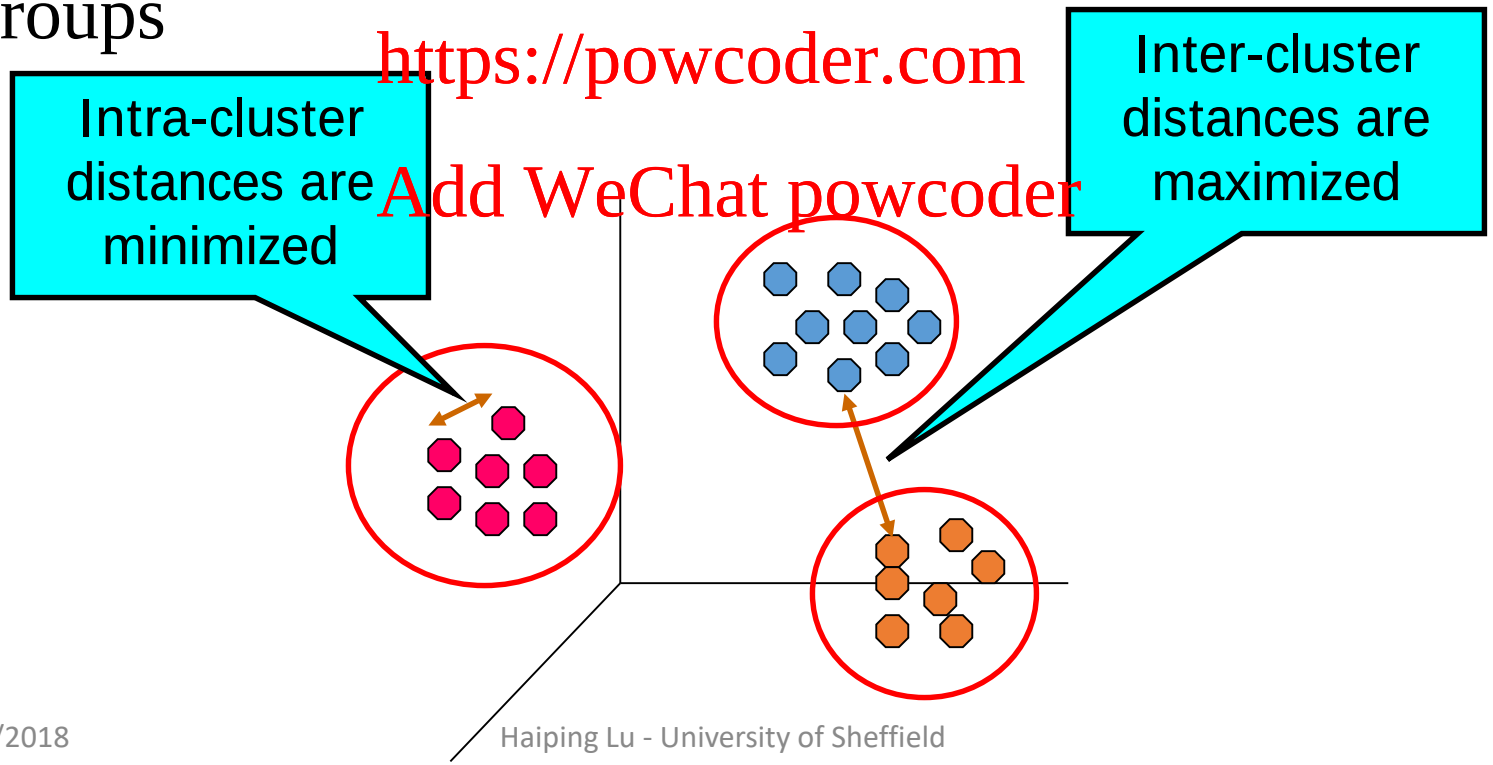
- K-means Clustering

- Scalable K-means

- Scalable K-means in Spark

What is Cluster Analysis?

- Finding groups of objects such that the objects in a group will be similar (or related) to one another and different from (or unrelated to) the objects in other groups



Cluster Analysis

- Divide data into (clusters) that are meaningful, useful, or both
- The study of techniques for automatically finding classes
- Clusters can help capture the natural structure of the data
- A starting point to further analysis
- An important role in a wide variety of fields:
  psychology, biology, statistics, pattern recognition, information retrieval, machine learning and data mining, etc

Clustering for Understanding

- Classes, or conceptually meaningful groups of objects that share some similarities, play an important role in how people analyze the describe the world
- Human beings are skilled at dividing objects into groups (clustering) and assigning particular objects to these groups (classification). E.g. children can quickly label the objects in a photograph as buildings, vehicles, people, animals, etc

Applications of Clustering

- Biology
  - Cluster analysis help create taxonomy of all living things: kingdom, phylum, class, order, family, etc
  - Cluster analysis on gene / protein data help annotate the function of genes / proteins
- Information retrieval.
  - Clustering help group the search results into a small number of clusters, each of which captures a particular aspect of the query. E.g. a query of “movie” might return Web pages grouped into categories such as reviews, trailers, starts, and theaters
- Climate
  - Cluster analysis has been applied to find patterns in the atmospheric pressure of polar regions and areas of the ocean that have a significant impact on land climate

Applications of Clustering

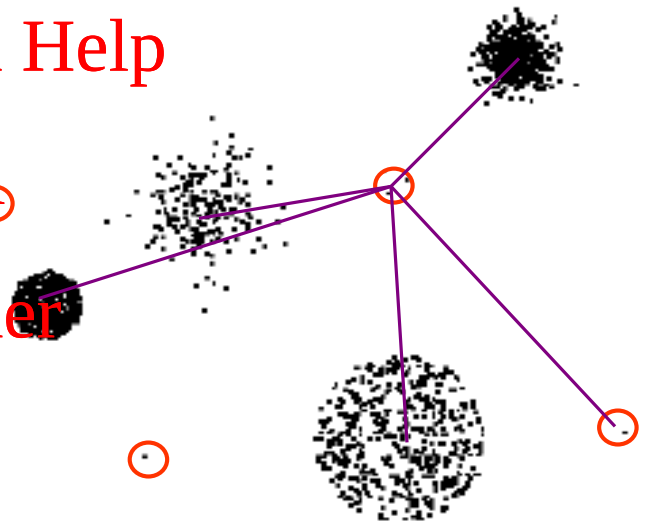
- Psychology and Medicine.
  - Identify different types of diseases (e.g. depression)
  - Detect patterns in the spatial or temporal distribution of a disease
  - Help group patients with similar patterns
- Business
  - Clustering analysis can be used to segment customers into a small number of groups for additional analysis and marketing activities
- **Anomaly/Outlier Detection (notebook/coursework data)**

Anomaly/Outlier Detection

- What are anomalies/outliers?
  - The set of data points that are considerably different than the remainder of the data
- Applications:
  - Credit card fraud detection: purchasing behavior
  - Network intrusion detection: unusual behavior
  - Ecosystem disturbances: typhoon, fire
  - Public health: SARS, bird flu, HxNx
  - Medicine: unusual symptoms/test results

Clustering-Based Anomaly/Outlier Detection

- Cluster the data into groups of different density
- Choose points in small cluster as candidate outliers
- Compute the distance between candidate points and non-candidate clusters.
- If candidate points are far from all other non-candidate points, they are outliers



About Cluster Analysis

- Cluster analysis groups data objects based only on information found in the data that describes the objects and their relationships
- The goal is that the objects within a group be similar (or related) to one another and different from (or unrelated to) the objects in other groups
- The greater the similarity (or homogeneity) within a group and the greater the difference between groups, the better or more distinct the clustering.

Week 9 Contents

- Introduction to Cluster Analysis
- **K-means Clustering**
- Scalable K-means
- Scalable K-means in Spark

K-means Clustering

- A prototype-based, partitional clustering approach
- Each cluster is associated with a centroid (centre point)
- Each point is assigned to the cluster with the closest centroid
- Number of clusters, K, must be specified before clustering

K-means Clustering

- **Input:**
  - A set $X=\{x_1, x_2, \ldots, x_n\}$ of n data points
  - Number of clusters k
- For a set $C=\{c_1, c_2, \ldots, c_k\}$ of cluster "centres" define:

$$\varphi_X(C) = \sum_{x \in X} d(x,C)$$

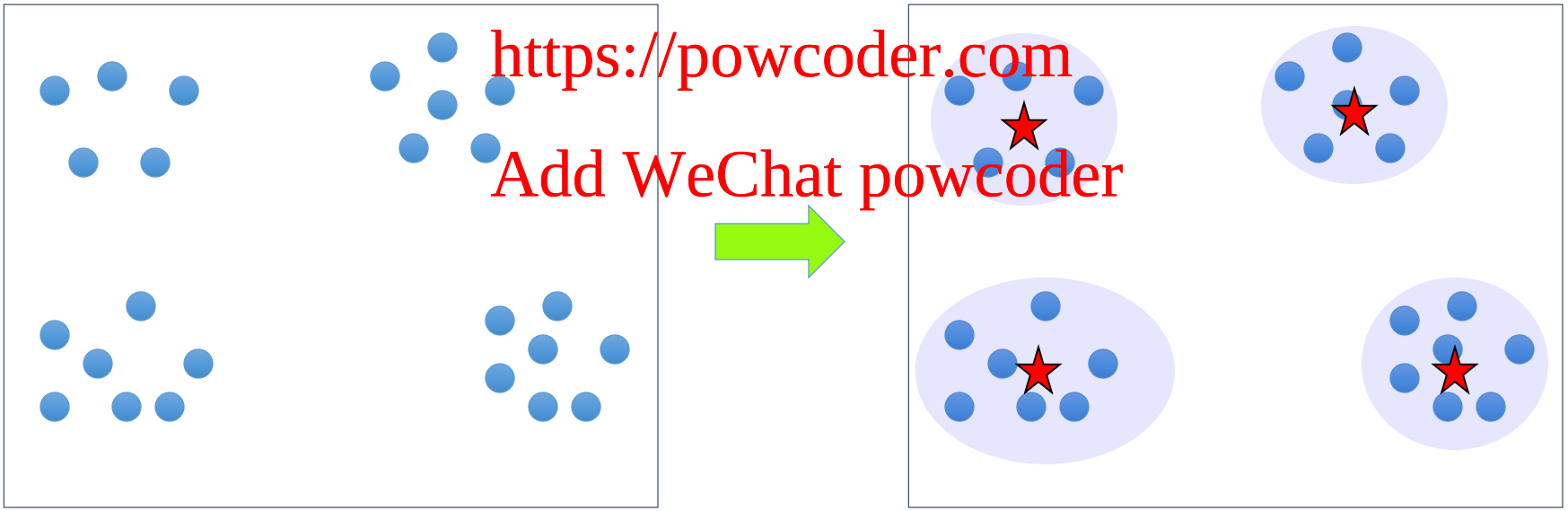
  where $d(x,C)$ = distance from x to closest centre in C
- **Goal:** To find a set C of centres that minimizes the objective function $\varphi_X(C)$

Determine the number of clusters

There are different approaches of determining K

- K can be arbitrarily set as any number
- K can be determined according to the need of further analysis
- K can be determined according to field knowledge, or the knowledge obtained during data visualisation
- Different K's can be initially set, and find the best K using some criteria

K-means Clustering: Example



K = 4

Lloyd Algorithm

- Start with k arbitrary centres $\{c_1, c_2, \ldots, c_k\}$ (typically chosen uniformly at random from data points)
- Performs an EM-type local search till convergence
- Main advantages: Simplicity, scalability (iterations)

1: Select K points as the initial centroids.
2: **repeat**
3: Form K clusters by assigning all points to the closest centroid.
4: Recompute the centroid of each cluster.
5: **until** The centroids don't change

What's wrong with Lloyd Algorithm?

- Takes many iterations to converge
- Very sensitive to initialization
- Random initialization can easily get two centres in the same cluster
  - K-means gets stuck in a local optimum

Lloyd Algorithm: Initialization

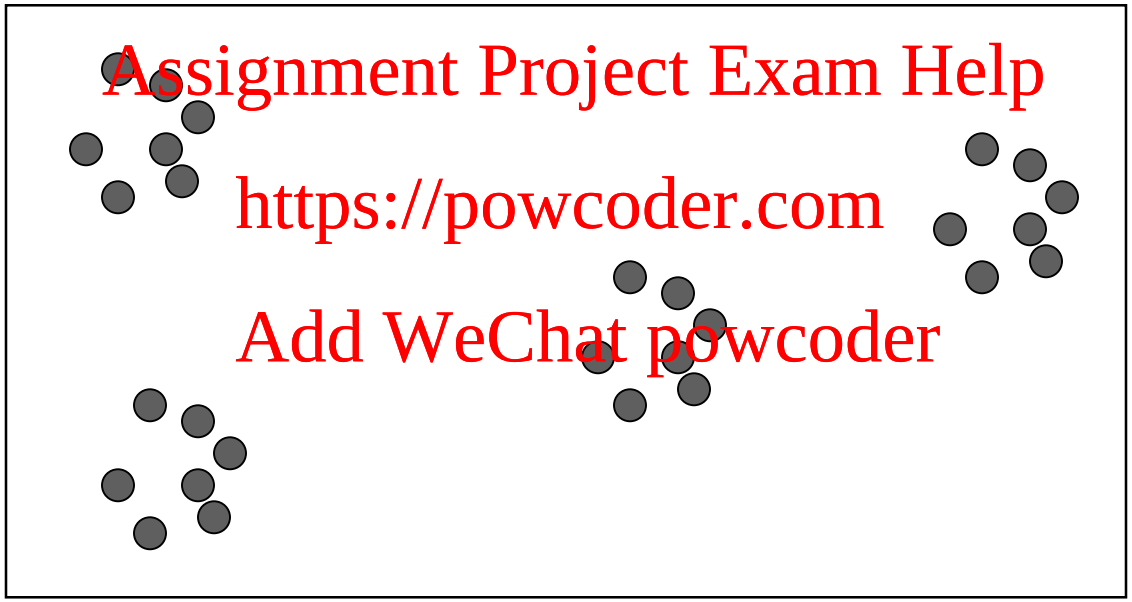


Figure credited to David Arthur

Lloyd Algorithm: Initialization

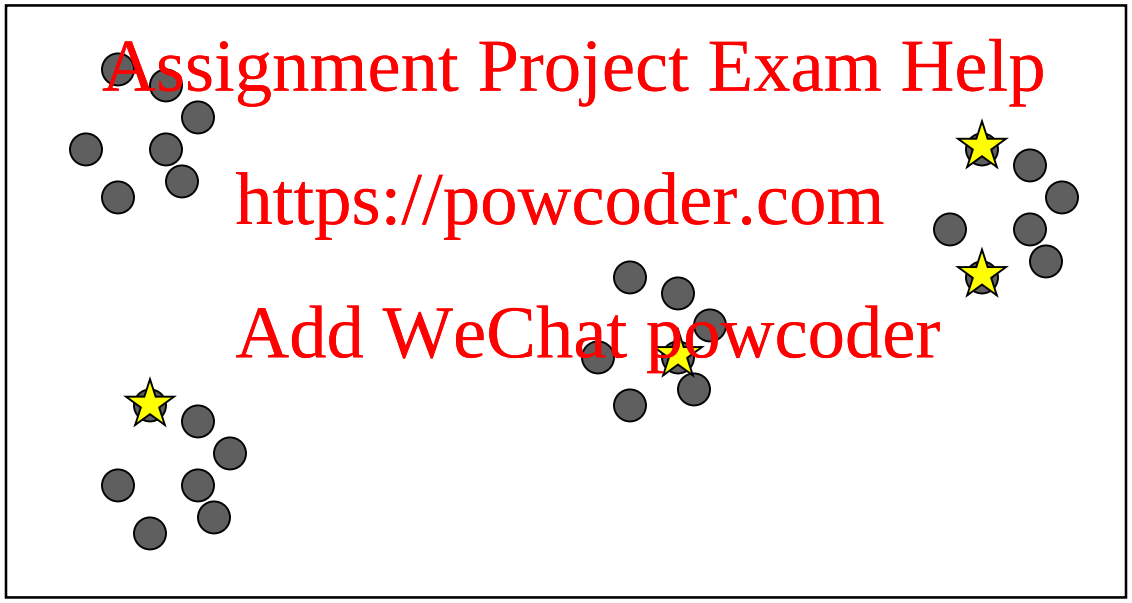


Figure credited to David Arthur

Lloyd Algorithm: Initialization

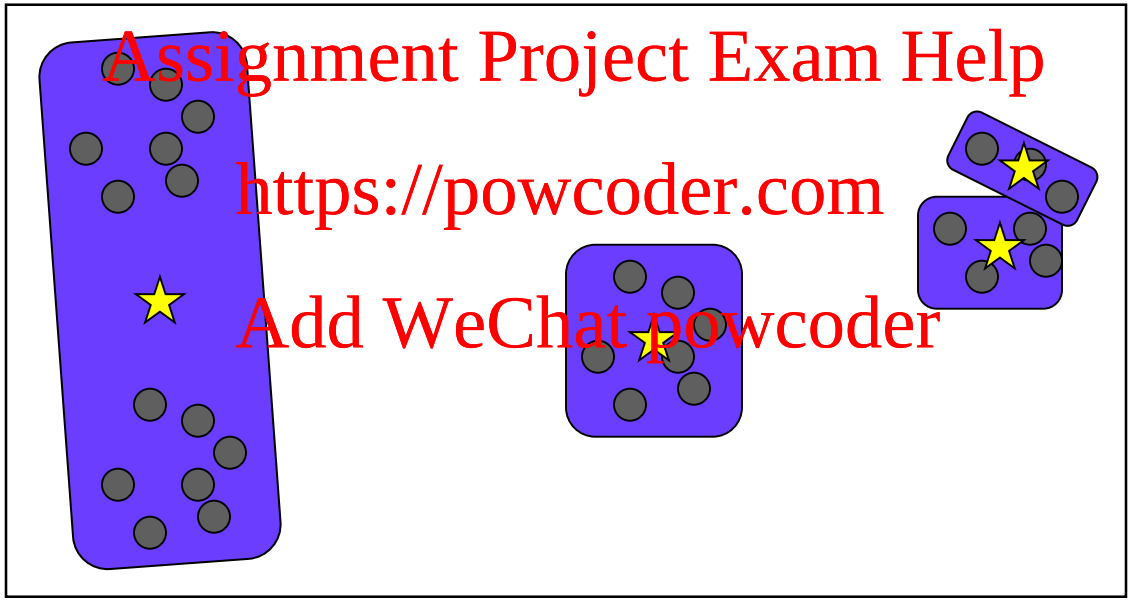


Figure credited to David Arthur

Lloyd Algorithm: Initialization

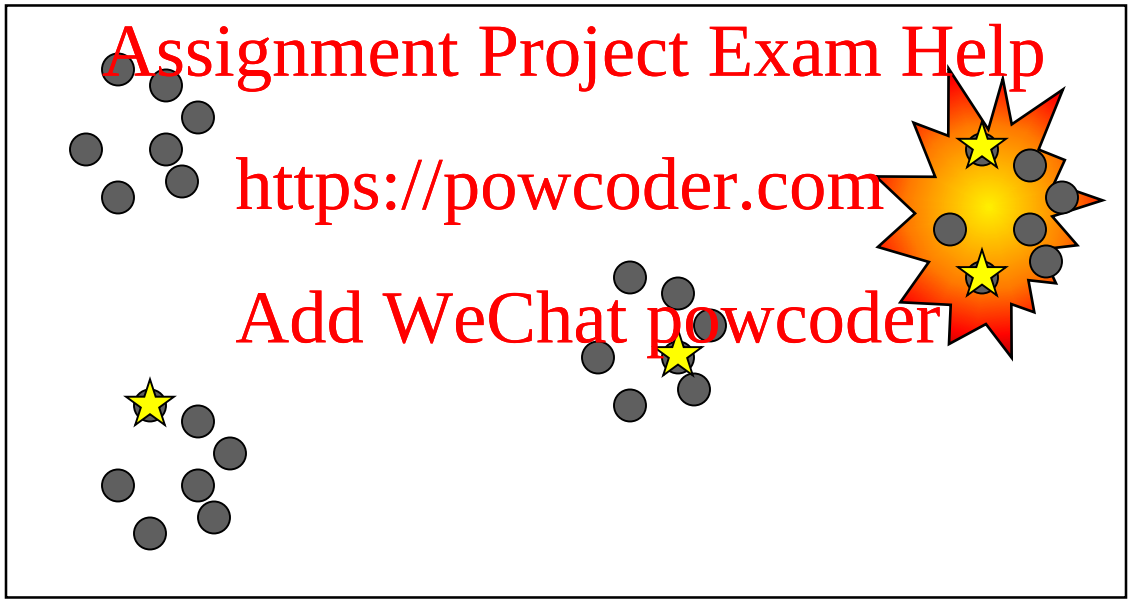


Figure credited to David Arthur

Week 9 Contents

- Introduction to Cluster Analysis
- K-means Clustering
- **Scalable K-means**
- Scalable K-means in Spark

K-means++ [Arthur et al. ’07]

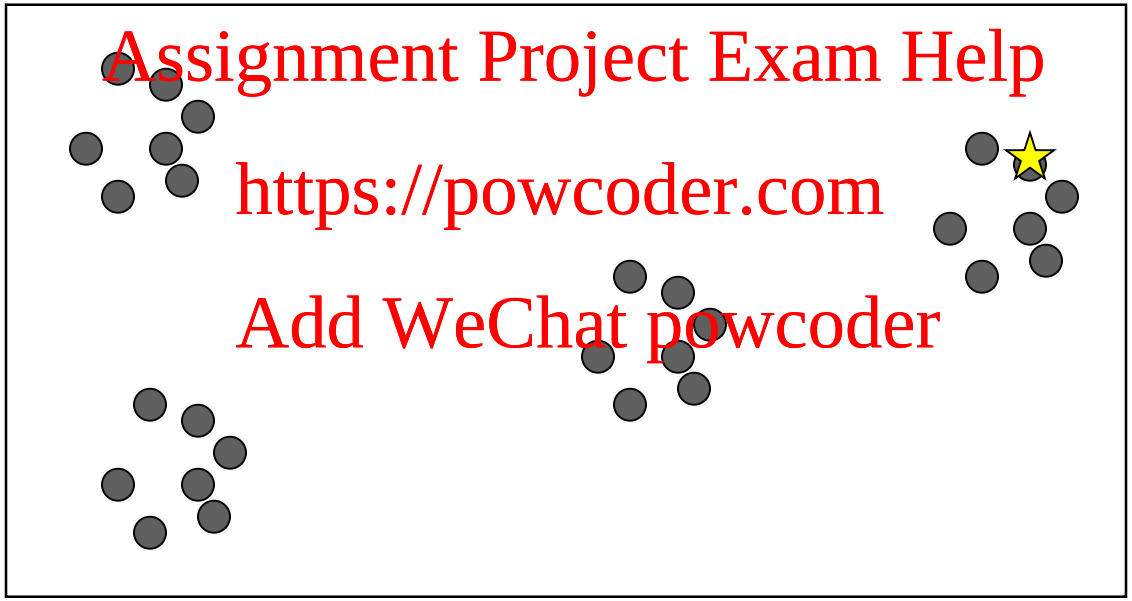
- Spreads out the centres
- Choose first centre c_1 uniformly at random from the data set
- Repeat for $2 \leq i \leq k$:
  - Choose c_i to be equal to a data point x_0 sampled from the distribution:

$$\frac{d(x_0, C)^2}{\varphi_X(C)} \propto d(x_0, C)^2$$

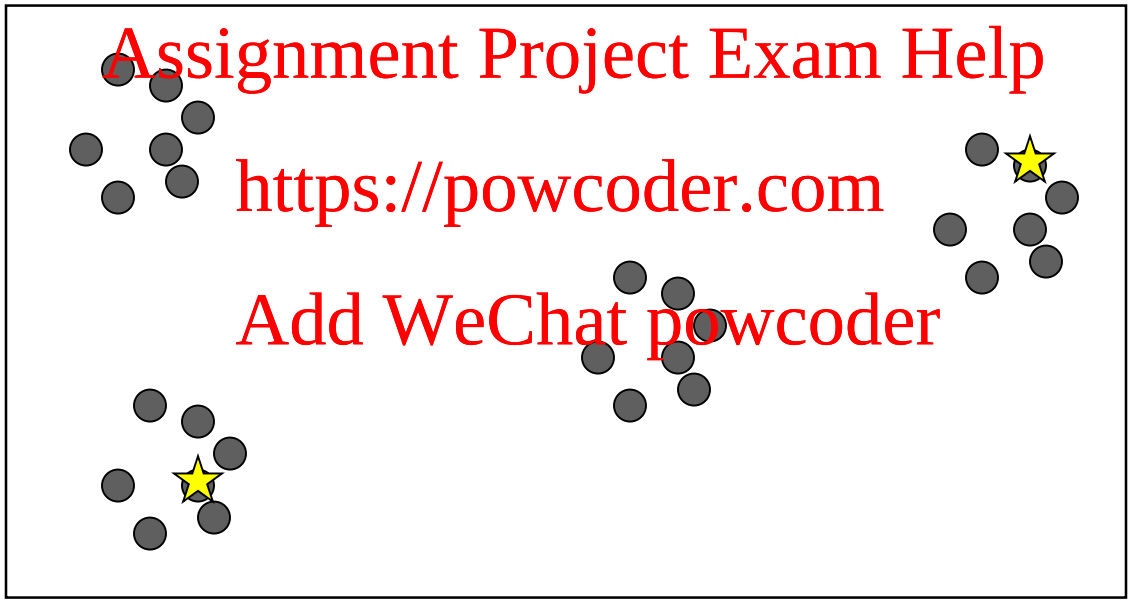
- **Theorem:** O(log k)-approximation to optimum, right after initialization

K-means++ Initialization

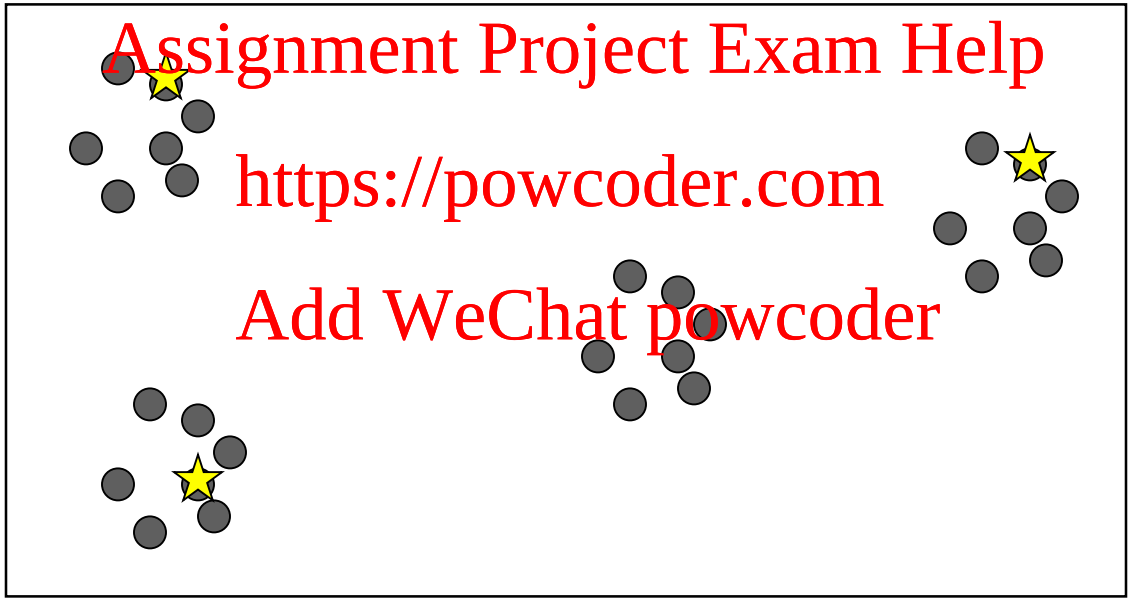
K-means++ Initialization



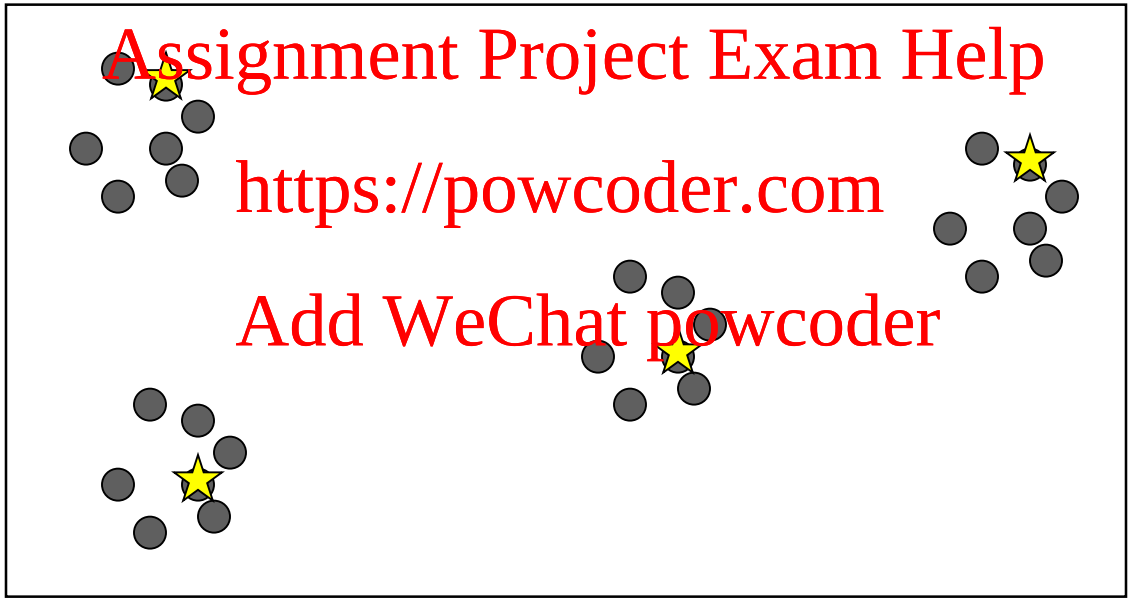
K-means++ Initialization



K-means++ Initialization



K-means++ Initialization



What's Wrong with K-means++?

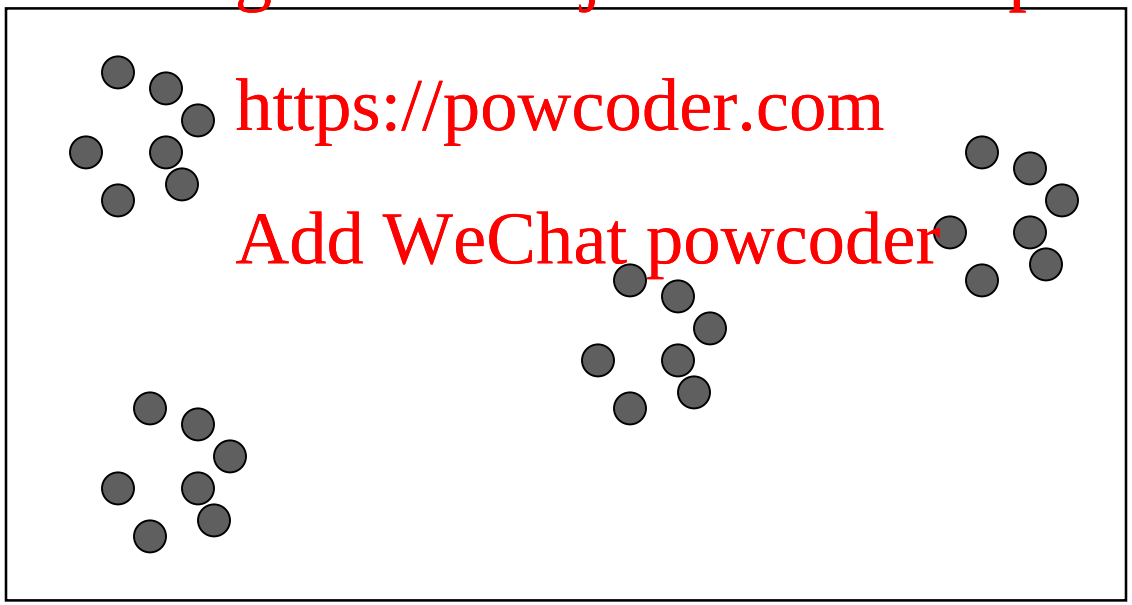
- Needs K passes over the data
- In large data applications, not only the data is massive, but also K is typically large (e.g., easily 1000).
- Does not scale!

Intuition for a Solution

- K-means++ samples one point per iteration and updates its distribution
- What if we **oversample** by sampling each point independently with a larger probability?
- Intuitively equivalent to updating the distribution much less frequently
  - Coarser sampling
- Turns out to be sufficient: K-means||

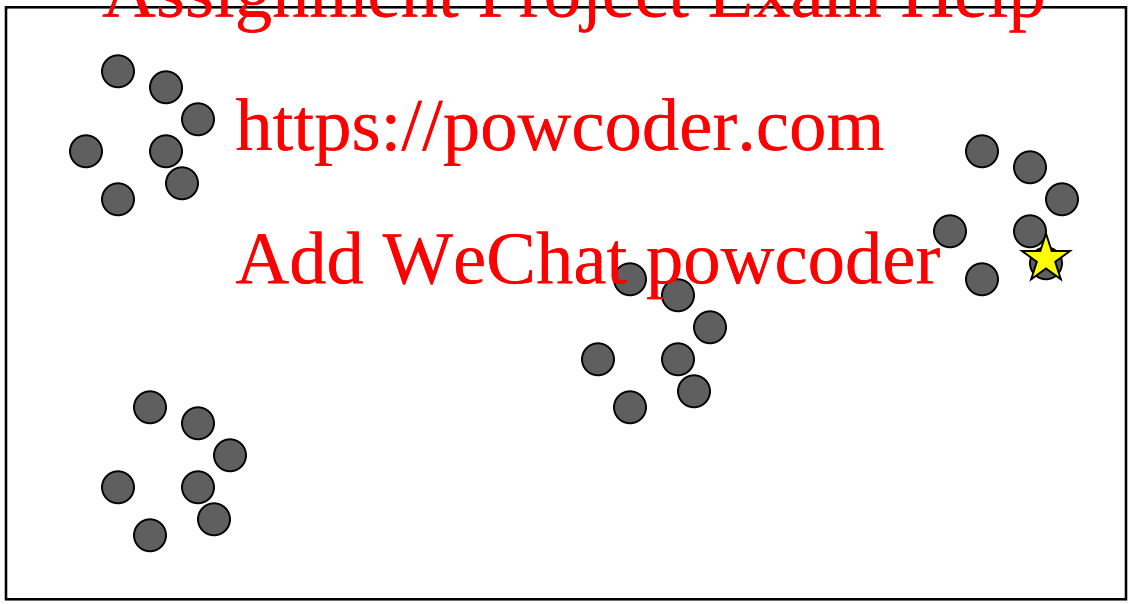
K-means‖ Initialization [Bahmani et al. ’12]

K=4,
Oversampling factor L=3



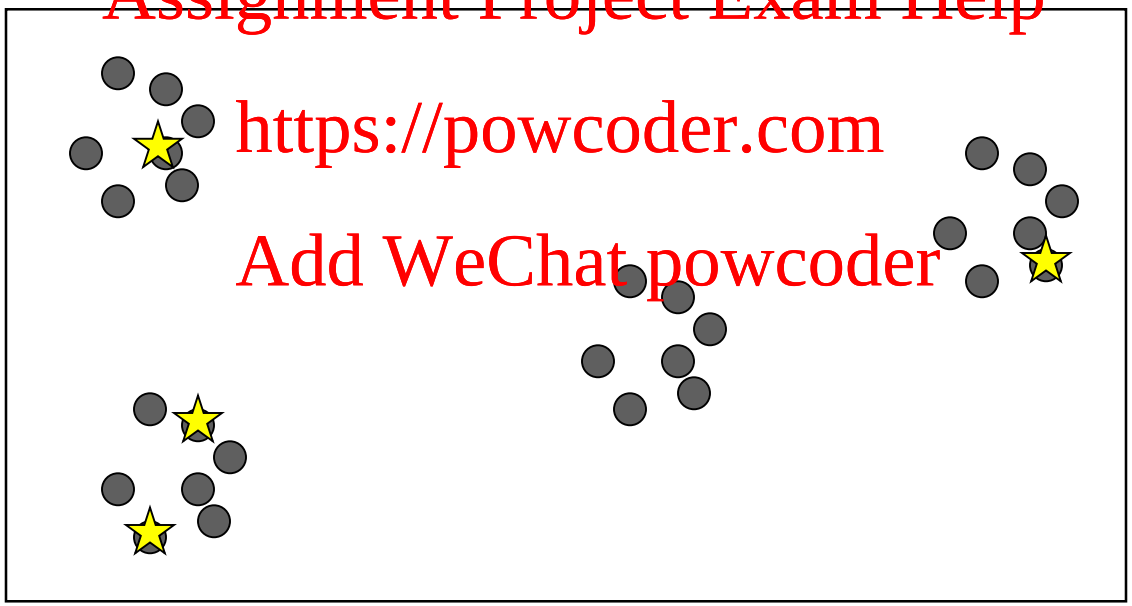
K-means|| Initialization

K=4,
Oversampling factor L=3



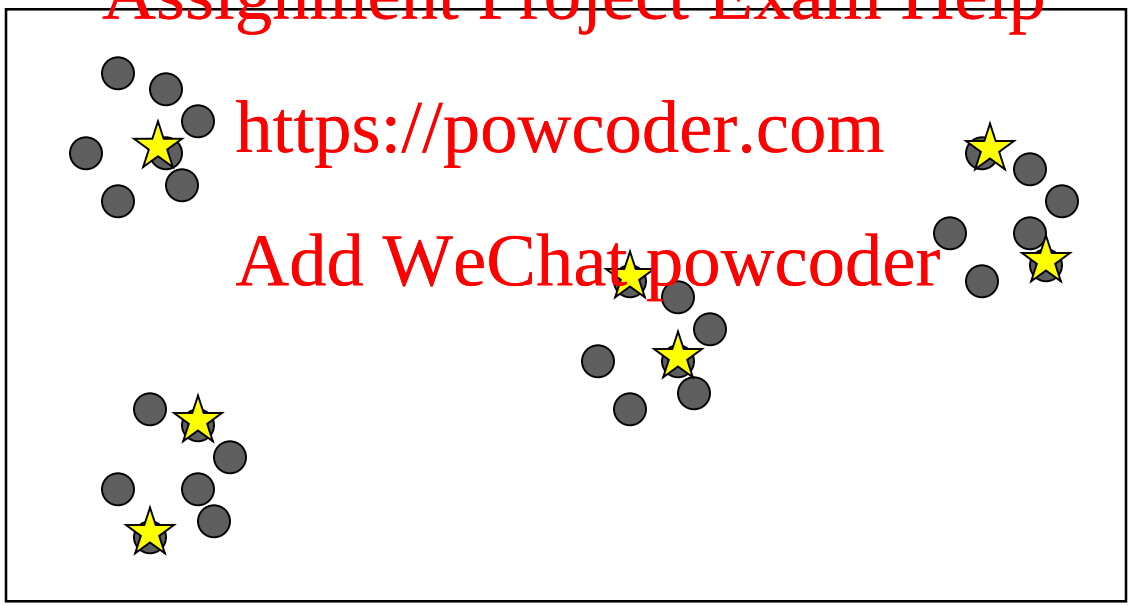
K-means|| Initialization

K=4,
Oversampling factor L=3



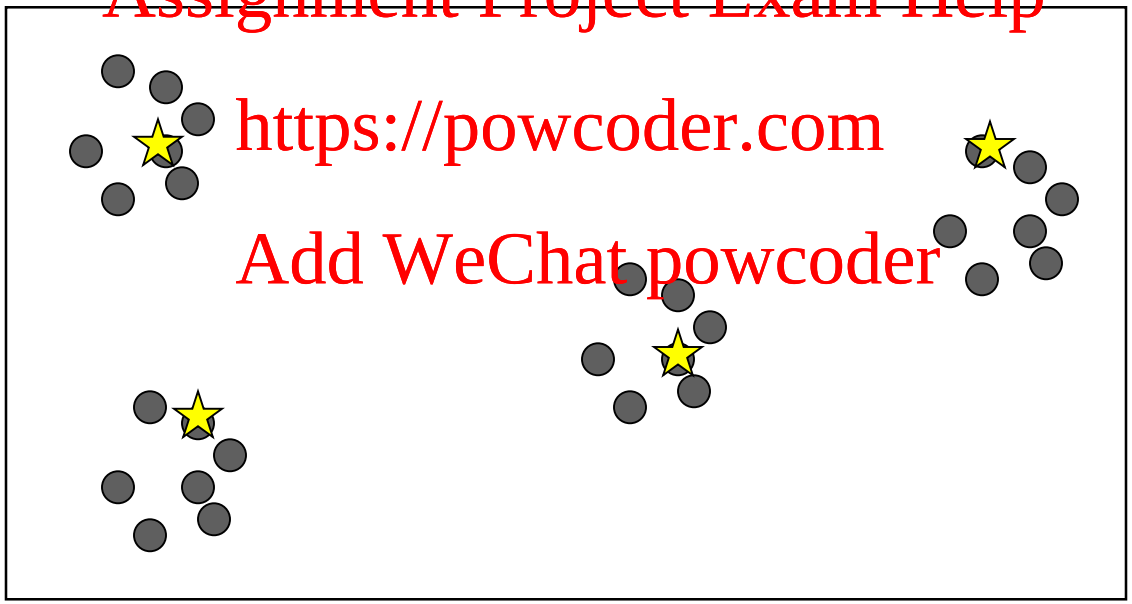
K-means|| Initialization

K=4,
Oversampling factor L=3



K-means|| Initialization

K=4,
Oversampling factor L=3



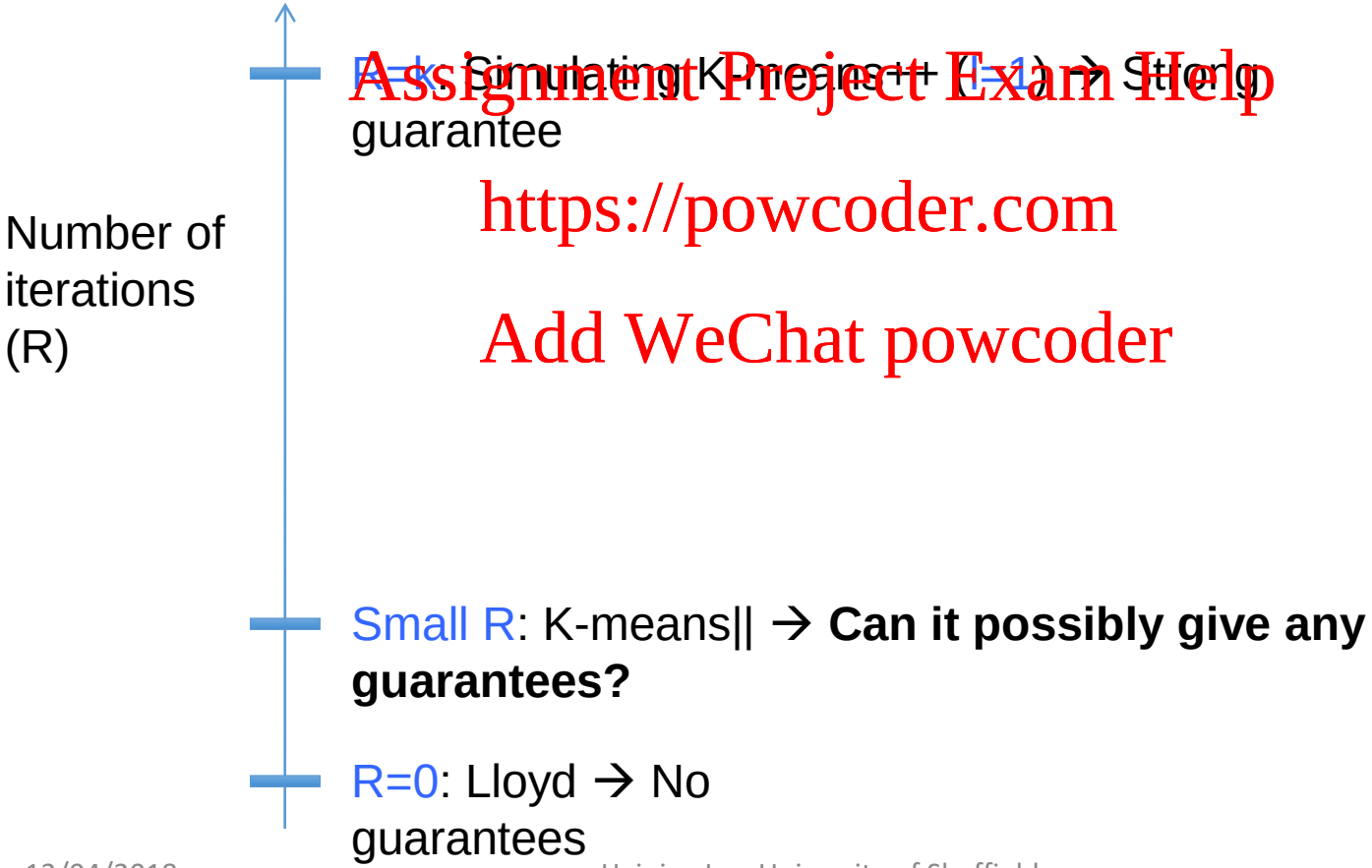
Cluster the intermediate centres

K-means|| [Bahmani et al. ’12]

- Choose $L>1$
- Initialize C to an arbitrary set of points
- For R iterations do:
  - Sample each point x in X independently with probability $p_x = Ld^2(x,C)/\phi_X(C)$
  - Add all the sampled points to C
- Cluster the (weighted) points in C to find the final k centres

K-means||: Intuition

- An interpolation between Lloyd and K-means++



K-means||: Benefits

- Using K-means++ for clustering the intermediate centres, the overall approximation factor = **O(log k)**
- K-means|| much harder than K-means++ to get confused with noisy outliers
- K-means|| reduces number of Lloyd iterations even more than K-means++

Week 9 Contents

- Introduction to Cluster Analysis
- K-means Clustering
- Scalable K-means
- **Scalable K-means in Spark**

K-means in MLlib (notebook)

- Not scalable: Kmeans
- Scalable: Kmeans | | (default)
- Code: https://github.com/apache/spark/blob/v2.1.0/mllib/src/main/scala/org/apache/spark/mllib/clustering/KMeans.scala
- Documentation: https://spark.apache.org/docs/2.1.0/api/scala/index.html#org.apache.spark.mllib.clustering.KMeans
- https://spark.apache.org/docs/2.1.0/mllib-clustering.html

K-means in Mllib (notebook)

- *k:* the number of desired clusters.
- *maxIterations:* the maximum number of iterations
- *initializationMode:* specifies either random initialization or initialization via k-means|| (compare)
- *runs*: no effect since Spark 2.0.0.
- *initializationSteps*: determines the number of steps in the k-means|| algorithm (default=2, advanced)
- *epsilon:* determines the distance threshold within which we consider k-means to have converged
- *initialModel:* manually set cluster centres for initialization

K-means in ML

- An Estimator
- Uses MLlib Kmeans (Kmeans | |)
- Code: https://github.com/apache/spark/blob/v2.1.0/mllib/src/main/scala/org/apache/spark/ml/clustering/KMeans.scala
- Documentation: https://spark.apache.org/docs/2.1.0/api/scala/index.html#org.apache.spark.ml.clustering.KMeans
- https://spark.apache.org/docs/2.1.0/ml-clustering.html

K-means in ML

- *k:* the number of desired clusters.
- *maxIter:* the maximum number of iterations
- *initMode:* specifies either random initialization or initialization via k-means|| (compare)
- *initSteps*: determines the number of steps in the k-means|| algorithm (default=2, advanced)
- *tol:* determines the distance threshold within which we consider k-means to have converged.
- *initialModel:* manually set cluster centres for initialization

Simplified from those for K-means in MLlib

Running Scalable K-means

- RDD should be cached for high performance (check warning when you run your program)


```
val centers = initialModel match {
  case Some(kMeansCenters) =>
    kMeansCenters.clusterCenters.map(new VectorWithNorm(_))
  case None =>
    if (initializationMode == KMeans.RANDOM) {
      initRandom(data)
    } else {
      initKMeansParallel(data)
    }
}
```

K-means++ in Spark

```
// Finally, we might have a set of more than k distinct candidate centers; weight each
// candidate by the number of points in the dataset mapping to it and run a local k-means++
// on the weighted centers to pick k of them
val bcCenters = data.context.broadcast(distinctCenters)
val countMap = data.map(KMeans.findClosest(bcCenters.value, _)._1).countByValue()

bcCenters.destroy(blocking = false)

val myWeights = distinctCenters.indices.map(countMap.getOrElse(_, 0L).toDouble).toArray
LocalKMeans.kMeansPlusPlus(0, distinctCenters.toArray, myWeights, k, 30)
```


- Code: https://github.com/apache/spark/blob/master/mllib/src/main/scala/org/apache/spark/mllib/clustering/LocalKMeans.scala

Remark

- Acknowledgement
  - Some slides are adapted from the K-means|| slides by Bahman Bahmani, Stanford University, 2012
- References
  - Chapter on clustering from a classic textbook (88 pages): https://www-users.cs.umn.edu/~kumar/dmbook/ch7_clustering.pdf
  - K-means overview: https://en.wikipedia.org/wiki/K-means%2B%2B
  - K-means ++ paper: http://ilpubs.stanford.edu:8090/778/1/2006-13.pdf
  - K-means || paper: http://dl.acm.org/citation.cfm?doid=2180912.2180915
  - Spark ML: https://spark.apache.org/docs/2.1.0/api/scala/index.html#org.apache.spark.ml.clustering.KMeans
  - Spark MLLib: https://spark.apache.org/docs/2.1.0/api/scala/index.html#org.apache.spark.mllib.clustering.KMeans