Clustering Data: Does Theory Help?

Ravi Kannan

December 10, 2013

Clustering

- Given n objects - divide (partition) them into k clusters. Each cluster should consist of “similar” or “close-by” objects.

Clustering

- Given n objects - divide (partition) them into k clusters. Each cluster should consist of “similar” or “close-by” objects.
- Often, each object is just a vector. [One component per “feature”. Many features.]

Clustering

- Given n objects - divide (partition) them into k clusters. Each cluster should consist of “similar” or “close-by” objects.
- Often, each object is just a vector. [One component per “feature”. Many features.]
- Measure either “similarity” between objects (eg. dot product) or distance (dissimilarity).

Clustering

- Given n objects - divide (partition) them into k clusters. Each cluster should consist of "similar" or "close-by" objects.
- Often, each object is just a vector. [One component per "feature". Many features.]
- Measure either "similarity" between objects (eg. dot product) or distance (dissimilarity).
- TCS, Th. OR Find an OPTIMAL $k-$ clustering which
  - Minimize Σ (distance of data point to its cluster center).

Clustering

- Given n objects - divide (partition) them into k clusters. Each cluster should consist of "similar" or "close-by" objects.
- Often, each object is just a vector. [One component per "feature". Many features.]
- Measure either "similarity" between objects (eg. dot product) or distance (dissimilarity).
- TCS, Th. OR Find an OPTIMAL $k-$ clustering which
  - Minimize Σ (distance of data point to its cluster center).
  - OR Max sum of similarities within clusters.

Clustering

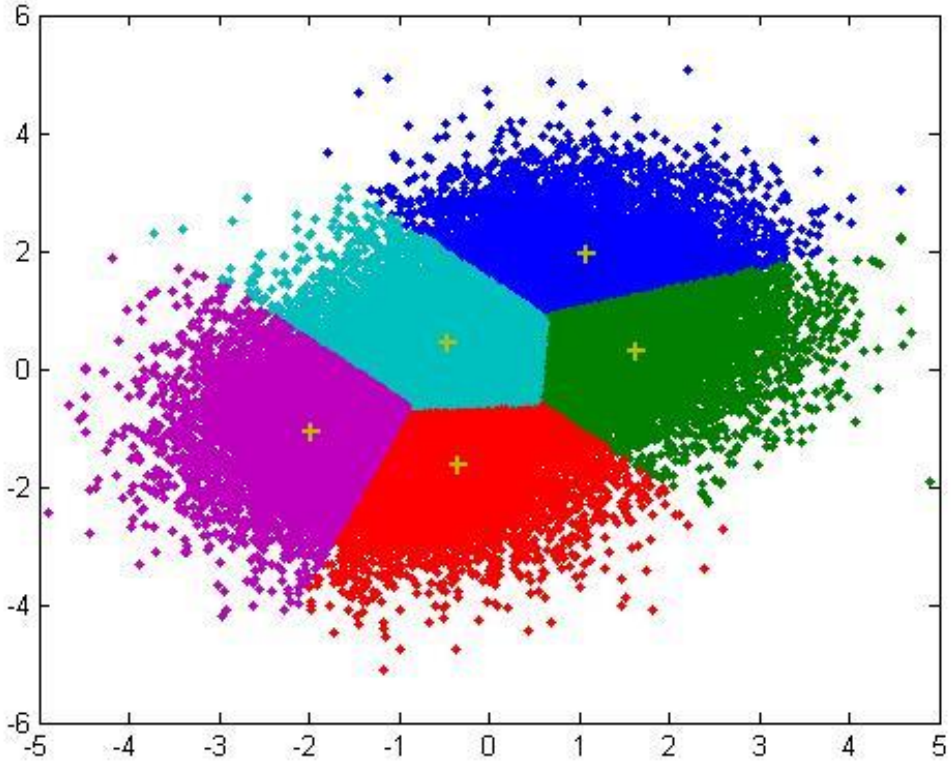
- Given n objects - divide (partition) them into k clusters. Each cluster should consist of "similar" or "close-by" objects.
- Often, each object is just a vector. [One component per "feature". Many features.]
- Measure either "similarity" between objects (eg. dot product) or distance (dissimilarity).
- TCS, Th. OR Find an OPTIMAL $k-$ clustering which
  - Minimize Σ (distance of data point to its cluster center).
  - OR Max sum of similarities within clusters.
- $k-$means Problem Minimize Sum of $(\text{Dist})^2$ to cluster centers.

Clustering

- Given n objects - divide (partition) them into k clusters. Each cluster should consist of "similar" or "close-by" objects.
- Often, each object is just a vector. [One component per "feature". Many features.]
- Measure either "similarity" between objects (eg. dot product) or distance (dissimilarity).
- TCS, Th. OR Find an OPTIMAL $k-$ clustering which
  - Minimize Σ (distance of data point to its cluster center).
  - OR Max sum of similarities within clusters.
- $k-$means Problem Minimize Sum of $(\text{Dist})^2$ to cluster centers.
- Many headaches of this talk (and the field) would be gone if we can exactly optimize. Alas Exact Optimization is NP-Hard, so can only approx optimize.

Picture

5 clusters



Ultimate Goal

- Find the CORRECT clustering.

Ultimate Goal

- Find the CORRECT clustering.
- Differences of view:

Ultimate Goal

- Find the CORRECT clustering.
- Differences of view:
  - TCS: The optimal Clustering is obviously the correct one. [Maybe right, but can't find THE optimal one.]

Ultimate Goal

- Find the CORRECT clustering.
- Differences of view:
  - TCS: The optimal Clustering is obviously the correct one. [Maybe right, but can't find THE optimal one.]
  - Statistics: The Correct clustering is the one used by the "invisible hand" to generate the data in the first place. [Stochastic Model of data - Prior.]

Ultimate Goal

- Find the CORRECT clustering.
- Differences of view:
  - TCS: The optimal Clustering is obviously the correct one. [Maybe right, but can't find THE optimal one.]
  - Statistics: The Correct clustering is the one used by the "invisible hand" to generate the data in the first place. [Stochastic Model of data - Prior.]
  - Practitioner: Give me your answer and I will tell you post facto whether it is the correct clustering.

Ultimate Goal

- Find the CORRECT clustering.
- Differences of view:
  - TCS: The optimal Clustering is obviously the correct one. [Maybe right, but can't find THE optimal one.]
  - Statistics: The Correct clustering is the one used by the "invisible hand" to generate the data in the first place. [Stochastic Model of data - Prior.]
  - Practitioner: Give me your answer and I will tell you post facto whether it is the correct clustering.

The Invisible Hand

TCS: Approximations for $k-$means Clustering

TCS: Approximations for $k-$means Clustering

TCS: Approximations for $k-$means Clustering

- Spectral Methods, yield solution with $k-$means at most constant times optimal.

TCS: Approximations for $k-$means Clustering

- Spectral Methods, yield solution with $k-$means at most constant times optimal.
- Theory at Work: Now ε OPT error algorithms available. Also simpler.

TCS: Approximations for $k-$means Clustering

- Spectral Methods, yield solution with $k-$means at most constant times optimal.
- Theory at Work: Now ε OPT error algorithms available. Also simpler.
- Kumar, Sabharwal, Sen: ε approximation in linear time when k is fixed. (Using ideas from Badiou, Har-Peled, Indyk; Inaba, Katoh, Imai; Matrousek).

TCS: Approximations for $k-$means Clustering

- Spectral Methods, yield solution with $k-$means at most constant times optimal.
- Theory at Work: Now ε OPT error algorithms available. Also simpler.
- Kumar, Sabharwal, Sen: ε approximation in linear time when k is fixed. (Using ideas from Badiou, Har-Peled, Indyk; Inaba, Katoh, Imai; Matrousek).
- Starting Idea: To get the center of one cluster: Centroid of small random sample from cluster is good enough.

TCS: Approximations for $k-$means Clustering

- Spectral Methods, yield solution with $k-$means at most constant times optimal.
- Theory at Work: Now ε OPT error algorithms available. Also simpler.
- Kumar, Sabharwal, Sen: ε approximation in linear time when k is fixed. (Using ideas from Badiou, Har-Peled, Indyk; Inaba, Katoh, Imai; Matrousek).
- Starting Idea: To get the center of one cluster: Centroid of small random sample from cluster is good enough.
- Random sample (of all data) contains subset from largest cluster.

TCS: Approximations for $k-$means Clustering

- Spectral Methods, yield solution with $k-$means at most constant times optimal.
- Theory at Work: Now ε OPT error algorithms available. Also simpler.
- Kumar, Sabharwal, Sen: ε approximation in linear time when k is fixed. (Using ideas from Badiou, Har-Peled, Indyk; Inaba, Katoh, Imai; Matrousek).
- Starting Idea: To get the center of one cluster: Centroid of small random sample from cluster is good enough.
- Random sample (of all data) contains subset from largest cluster.
  - Try all subsets. Peel off cluster "close to" centroid of subset. Repeat.

TCS: Approximations for $k-$means Clustering

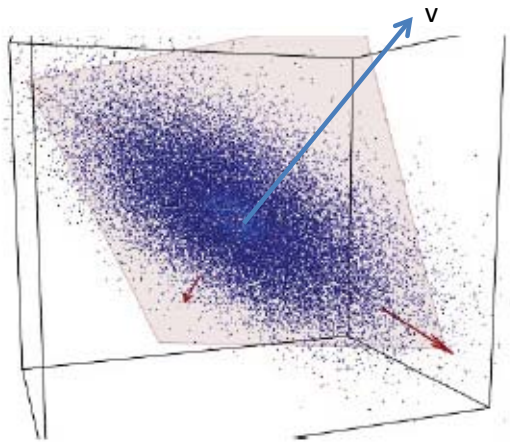
- Spectral Methods, yield solution with $k-$means at most constant times optimal.
- Theory at Work: Now ε OPT error algorithms available. Also simpler.
- Kumar, Sabharwal, Sen: ε approximation in linear time when k is fixed. (Using ideas from Badiou, Har-Peled, Indyk; Inaba, Katoh, Imai; Matrousek).
- Starting Idea: To get the center of one cluster: Centroid of small random sample from cluster is good enough.
- Random sample (of all data) contains subset from largest cluster.
  - Try all subsets. Peel off cluster "close to" centroid of subset. Repeat.
- Cannot go beyond constant size subsets, so cannot beat constant (small) factor error.

Spectral Clustering

- $A_1, A_2, \ldots, A_n$ are the data points to be clustered.

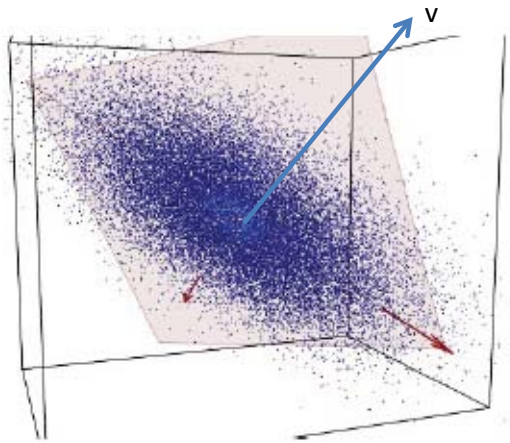
Spectral Clustering

- $A_1, A_2, \ldots, A_n$ are the data points to be clustered.
- $k-$means: Find k cluster centers. Set $C_1, C_2, \ldots, C_n$ each to be one of the k centers so as to minimize
  Sum of Squared Distances to A_i to C_i.



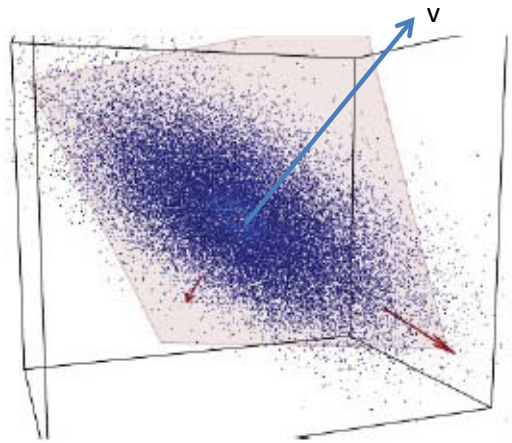
Spectral Clustering

- $A_1, A_2, \ldots, A_n$ are the data points to be clustered.
- $k-$means: Find k cluster centers. Set $C_1, C_2, \ldots, C_n$ each to be one of the k centers so as to minimize Sum of Squared Distances to A_i to C_i.
- Relax to rank$(C) \leq k$ instead of k distinct rows.



Spectral Clustering

- $A_1, A_2, \ldots, A_n$ are the data points to be clustered.
- $k-$means: Find k cluster centers. Set $C_1, C_2, \ldots, C_n$ each to be one of the k centers so as to minimize Sum of Squared Distances to A_i to C_i.
- Relax to rank$(C) \leq k$ instead of k distinct rows.
- Then, space spanned by C_i is the least-squares-fit $k-$dimensional space to $A_1, A_2, \ldots, A_n$. It can be found by Singular Value Decomposition. Principal Component Analysis - PCA.



PCA to Clustering

- Instead of finding the $k-$cluster centers minimizing sum of distance squared to cluster centers, PCA found for us the $k-$ dim'l subspace minimizing the sum of distances squared.

PCA to Clustering

- Instead of finding the $k-$cluster centers minimizing sum of distance squared to cluster centers, PCA found for us the $k-$ dim'l subspace minimizing the sum of distances squared.
- Natural Next step - project to this sub-space and find (approximately) optimal $k-$means clustering in subspace. Folklore: Spectral Clustering. Does it work ?

PCA to Clustering

- Instead of finding the $k-$cluster centers minimizing sum of distance squared to cluster centers, PCA found for us the $k-$ dim'l subspace minimizing the sum of distances squared.
- Natural Next step - project to this sub-space and find (approximately) optimal $k-$means clustering in subspace. Folklore: Spectral Clustering. Does it work ?
- yes, provably under stochastic models and more recently even under no stochastic assumptions...

PCA to Clustering

- Instead of finding the $k-$cluster centers minimizing sum of distance squared to cluster centers, PCA found for us the $k-$ dim'l subspace minimizing the sum of distances squared.
- Natural Next step - project to this sub-space and find (approximately) optimal $k-$means clustering in subspace. Folklore: Spectral Clustering. Does it work ?
- yes, provably under stochastic models and more recently even under no stochastic assumptions...
- First: The glories of PCA.

PCA to Clustering

- Instead of finding the $k-$cluster centers minimizing sum of distance squared to cluster centers, PCA found for us the $k-$ dim'l subspace minimizing the sum of distances squared.
- Natural Next step - project to this sub-space and find (approximately) optimal $k-$means clustering in subspace. Folklore: Spectral Clustering. Does it work ?
- yes, provably under stochastic models and more recently even under no stochastic assumptions...
- First: The glories of PCA.
- There is also a different way to do spectral clustering- by repeatedly using a 1-d projections, Fiedler; Shi, Malik; ... which we do not discuss here.

PCA and Noise

- Suppose $\underbrace{A}_{\text{Given}} = \underbrace{B}_{\text{Noise}} + \underbrace{C}_{\text{Data}}$, $\text{rank}(C) = k$.

PCA and Noise

- Suppose $\underbrace{A}_{\text{Given}} = \underbrace{B}_{\text{Noise}} + \underbrace{C}_{\text{Data}}$, $\text{rank}(C) = k$.
- Projection $\bar{A}$ of A to $k-$dim'l principal subspace is close to C (Folklore).

PCA and Noise

- Suppose $\underbrace{A}_{\text{Given}} = \underbrace{B}_{\text{Noise}} + \underbrace{C}_{\text{Data}}$, $\text{rank}(C) = k$.
- Projection $\bar{A}$ of A to $k-$dim'l principal subspace is close to C (Folklore).

PCA and Noise

- Suppose $\underbrace{A}_{\text{Given}} = \underbrace{B}_{\text{Noise}} + \underbrace{C}_{\text{Data}}$, $\text{rank}(C) = k$.
- Projection $\bar{A}$ of A to $k-$dim'l principal subspace is close to C (Folklore).

PCA and Noise

- Suppose $\underbrace{A}_{\text{Given}} = \underbrace{B}_{\text{Noise}} + \underbrace{C}_{\text{Data}}$, $\text{rank}(C) = k$.
- Projection $\bar{A}$ of A to $k-$dim'l principal subspace is close to C (Folklore).
- For a moment, assume noise $B_1, B_2, \ldots, B_n$ is "roughly equally spread in all directions".

PCA and Noise

- Suppose $\underbrace{A}_{\text{Given}} = \underbrace{B}_{\text{Noise}} + \underbrace{C}_{\text{Data}}$, $\text{rank}(C) = k$.
- Projection $\bar{A}$ of A to $k-$dim'l principal subspace is close to C (Folklore).
- For a moment, assume noise $B_1, B_2, \ldots, B_n$ is "roughly equally spread in all directions".
- Then, Error of PCA is at most $\frac{8k}{d}$ Total Noise.

PCA and Noise

- Suppose $\underbrace{A}_{\text{Given}} = \underbrace{B}_{\text{Noise}} + \underbrace{C}_{\text{Data}}$, $\text{rank}(C) = k$.
- Projection $\bar{A}$ of A to $k-$dim'l principal subspace is close to C (Folklore).
- For a moment, assume noise $B_1, B_2, \ldots, B_n$ is "roughly equally spread in all directions".
- Then, Error of PCA is at most $\frac{8k}{d}$ Total Noise.
- $\sum_i |\bar{A}_i - C_i|^2 \leq \frac{8k}{d} \sum_i |B_i|^2$.

PCA and Noise

- Suppose $\underbrace{A}_{\text{Given}} = \underbrace{B}_{\text{Noise}} + \underbrace{C}_{\text{Data}}$, $\text{rank}(C) = k$.
- Projection $\bar{A}$ of A to $k-$dim'l principal subspace is close to C (Folklore).
- For a moment, assume noise $B_1, B_2, \ldots, B_n$ is "roughly equally spread in all directions".
- Then, Error of PCA is at most $\frac{8k}{d}$ Total Noise.
- $\sum_i |\bar{A}_i - C_i|^2 \leq \frac{8k}{d} \sum_i |B_i|^2$.
- Big gain if $k << d$.

Denoising -formally

- Often, denoising is formally stated and proved with a bunch of stochastic assumptions. But really it is a simple lemma with no assumptions with a 5 line proof. (An “exercise” to prove, but perhaps took us long to formulate the clean, general statement.)

Denoising -formally

- Often, denoising is formally stated and proved with a bunch of stochastic assumptions. But really it is a simple lemma with no assumptions with a 5 line proof. (An "exercise" to prove, but perhaps took us long to formulate the clean, general statement.)
- Achlioptas, McSherry,.....,Hopcroft, Kannan: Simple Denoising Lemma A any matrix. $\bar{A}$ projection of A to $k-$dim principal subspace. C any matrix of rank k.
  $||\bar{A} - C||_F^2 \leq 8k||A - C||_2^2$,
  where, $||\cdot||_F^2$ is sum of squares of all entries and $||X||_2 = \text{Max}|Xu|$, over all unit vectors u.

Denoising -formally

- Often, denoising is formally stated and proved with a bunch of stochastic assumptions. But really it is a simple lemma with no assumptions with a 5 line proof. (An "exercise" to prove, but perhaps took us long to formulate the clean, general statement.)
- Achlioptas, McSherry,.....,Hopcroft, Kannan: Simple Denoising Lemma A any matrix. $\bar{A}$ projection of A to $k-$dim principal subspace. C any matrix of rank k.
  $||\bar{A} - C||_F^2 \leq 8k||A - C||_2^2$,
  where, $||\cdot||_F^2$ is sum of squares of all entries and $||X||_2 = \text{Max}|Xu|$, over all unit vectors u.
- lhs: $d-$dim's distances. rhs: 1-dim's distances.

Denoising -formally

- Often, denoising is formally stated and proved with a bunch of stochastic assumptions. But really it is a simple lemma with no assumptions with a 5 line proof. (An "exercise" to prove, but perhaps took us long to formulate the clean, general statement.)
- Achlioptas, McSherry,.....,Hopcroft, Kannan: Simple Denoising Lemma A any matrix. $\bar{A}$ projection of A to $k-$dim principal subspace. C any matrix of rank k.
  $||\bar{A} - C||_F^2 \leq 8k||A - C||_2^2$,
  where, $||\cdot||_F^2$ is sum of squares of all entries and $||X||_2 = \text{Max}|Xu|$, over all unit vectors u.
- lhs: $d-$dim's distances. rhs: 1-dim's distances.
- One $\bar{A}$ is close to EVERY C !!!

Project and Cluster

- Project data points to space spanned by top k singular vectors (PCA).

Project and Cluster

- Project data points to space spanned by top k singular vectors (PCA).
- Do an approximate clustering in the projection with thanks to TCS.

Project and Cluster

- Project data points to space spanned by top k singular vectors (PCA).
- Do an approximate clustering in the projection with thanks to TCS.
- If data was generated by a mixture model (of spherical gaussians), then this does the job.

Project and Cluster

- Project data points to space spanned by top k singular vectors (PCA).
- Do an approximate clustering in the projection with thanks to TCS.
- If data was generated by a mixture model (of spherical gaussians), then this does the job.
- Indeed, even if the data was not generated from a mixture model, will see that this provides a good start for $k-$means (from which we get rapid convergence).

Algorithms for PCA - Theory to Practice

- Can we sub-sample a large matrix to pick a few rows/columns of it, do PCA on the sub-matrix and infer anything about PCA on the whole matrix?

Algorithms for PCA - Theory to Practice

- Can we sub-sample a large matrix to pick a few rows/columns of it, do PCA on the sub-matrix and infer anything about PCA on the whole matrix?
- Frieze, Kannan, Vempala (1995): Yes if we pick rows/columns with probability proportional to squared length. But not practical.

Algorithms for PCA - Theory to Practice

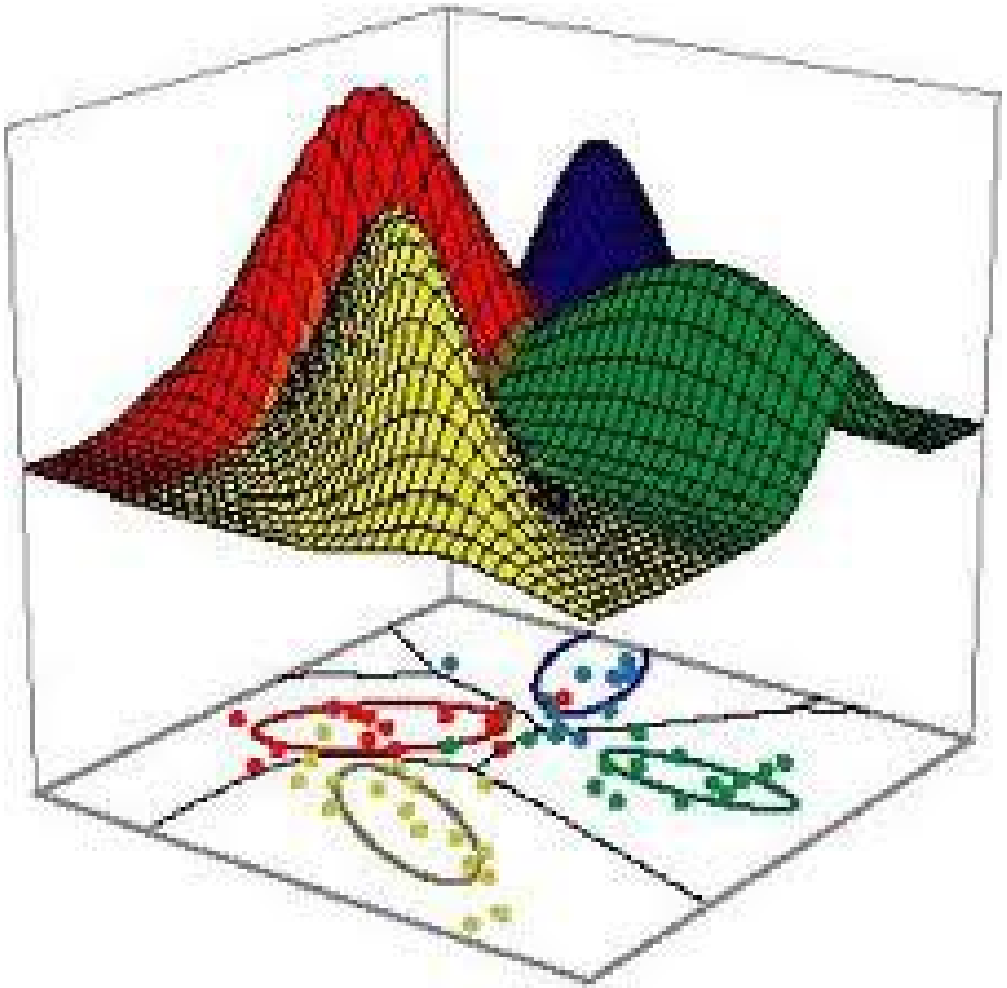
- Can we sub-sample a large matrix to pick a few rows/columns of it, do PCA on the sub-matrix and infer anything about PCA on the whole matrix?
- Frieze, Kannan, Vempala (1995): Yes if we pick rows/columns with probability proportional to squared length. But not practical.
- Decade of development by Drineas, Mahoney, many others... Presented in Workshops this semester. "Sampling based methods are now a crucial ingredient of computing with large matrices."
- Clarkson, Woodruff Nearly best rank k approx to A can be found in time linear in the number of non-zero entries in A if $k \in O(1)$.

Algorithms for PCA - Theory to Practice

- Can we sub-sample a large matrix to pick a few rows/columns of it, do PCA on the sub-matrix and infer anything about PCA on the whole matrix?
- Frieze, Kannan, Vempala (1995): Yes if we pick rows/columns with probability proportional to squared length. But not practical.
- Decade of development by Drineas, Mahoney, many others... Presented in Workshops this semester. "Sampling based methods are now a crucial ingredient of computing with large matrices."
- Clarkson, Woodruff Nearly best rank k approx to A can be found in time linear in the number of non-zero entries in A if $k \in O(1)$.
- Using Subspace Embeddings: If S is a $r \times n$ matrix ($r << n$) with one ± 1 entry per column chosen at random, then simultaneously for all vectors x,
  $|SAx|$ is within relative error ε of $|Ax|$

Mixture Models

Stochastic Model of data for clustering problems.



Mixture Models-II

- Probability Density F on $d-$space.

Mixture Models-II

- Probability Density F on $d-$space.
- F is a mixture of k components.
  $F = w_1 F_1 + w_2 F_2 + \cdots + w_k F_k$,
  each F_i is a Gaussian, say and $w_1, w_2, \ldots, w_k$ nonnegative summing to 1.

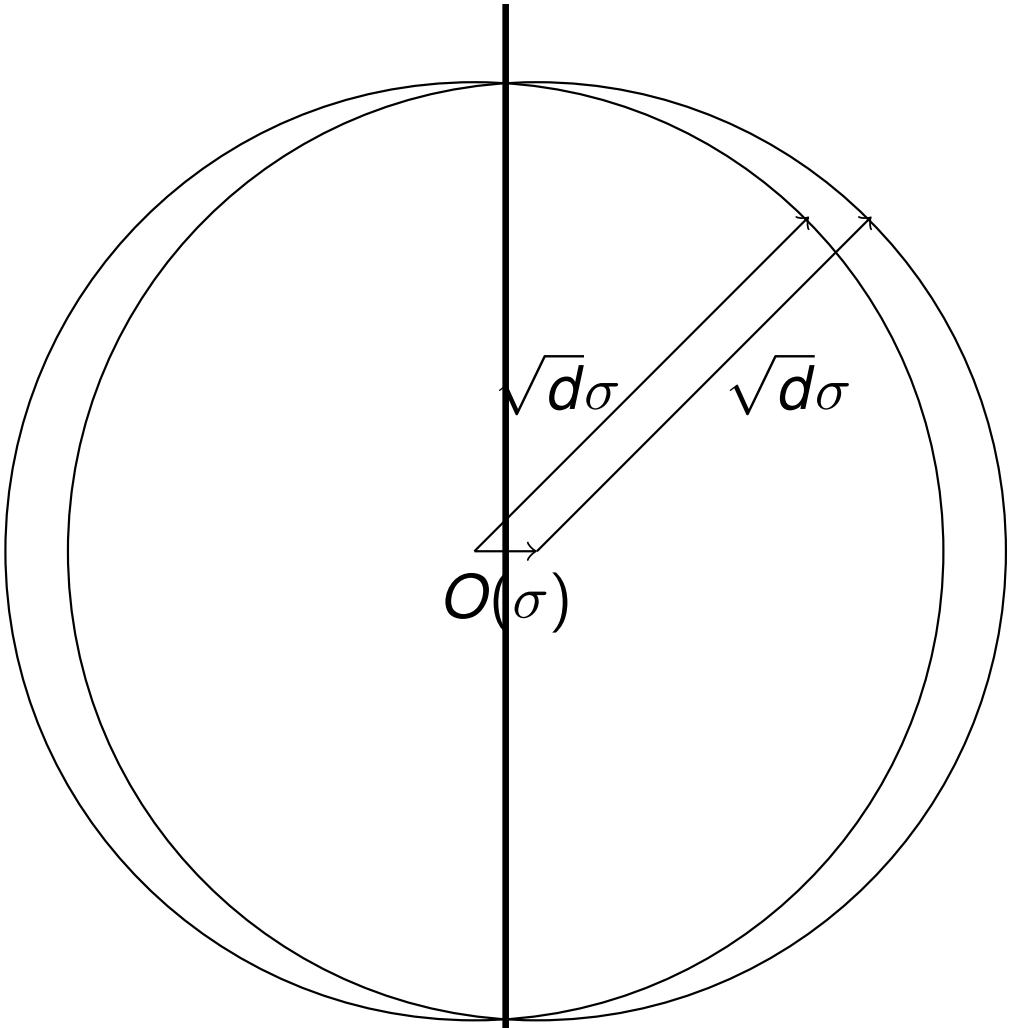
Mixture Models-II

- Probability Density F on $d-$space.
- F is a mixture of k components.
  $F = w_1 F_1 + w_2 F_2 + \cdots + w_k F_k$,
  each F_i is a Gaussian, say and $w_1, w_2, \ldots, w_k$ nonnegative summing to 1.
- Data points are n i.i.d. samples, each drawn according to F.

Mixture Models-II

- Probability Density F on $d-$space.
- F is a mixture of k components.
  $F = w_1 F_1 + w_2 F_2 + \cdots + w_k F_k$,
  each F_i is a Gaussian, say and $w_1, w_2, \ldots, w_k$ nonnegative summing to 1.
- Data points are n i.i.d. samples, each drawn according to F.
- Given data points, cluster them into k clusters corresponding to $F_1, F_2, \ldots, F_k$. Then it is easy fit a Gaussian to each cluster.

Two Gaussians of S.D. σ



Pts of Gaussian 1 $\leftrightarrow$ Pts of Gaussian 2

Typical picture in d dimensions

MSD is $O(d\sigma^2)$

Inter-center Sep is $O(\sigma)$

Error of $\varepsilon d\sigma^2$
Can mis-cluster many points !!

Correct Unit: σ

Correct Units in space of centers

***PICTURE OF TWO GAUSSIANS WITH PROJECTION

Inter-Center Distance and Clustering

- In 1-dimension, if we have two Gaussians of standard deviation (S.D.) 1 each:

Inter-Center Distance and Clustering

- In 1-dimension, if we have two Gaussians of standard deviation (S.D.) 1 each:
  - If inter-center separation is at least 100 S.D.'s, we can tell them apart.

Inter-Center Distance and Clustering

- In 1-dimension, if we have two Gaussians of standard deviation (S.D.) 1 each:
  - If inter-center separation is at least 100 S.D.'s, we can tell them apart.
  - If inter-center separation is less than $1/100$, difficult to tell them apart.

Inter-Center Distance and Clustering

- In 1-dimension, if we have two Gaussians of standard deviation (S.D.) 1 each:
  - If inter-center separation is at least 100 S.D.'s, we can tell them apart.
  - If inter-center separation is less than $1/100$, difficult to tell them apart.
- In $d-$ dimensions, a similar result holds Vempala, Wang:

Inter-Center Distance and Clustering

- In 1-dimension, if we have two Gaussians of standard deviation (S.D.) 1 each:
  - If inter-center separation is at least 100 S.D.'s, we can tell them apart.
  - If inter-center separation is less than $1/100$, difficult to tell them apart.
- In $d-$ dimensions, a similar result holds Vempala, Wang:
  - $k = O(1)$ spherical gaussian components of S.D 1 each. Inter-center separation of at least a constant*.

Inter-Center Distance and Clustering

- In 1-dimension, if we have two Gaussians of standard deviation (S.D.) 1 each:
  - If inter-center separation is at least 100 S.D.'s, we can tell them apart.
  - If inter-center separation is less than $1/100$, difficult to tell them apart.
- In $d-$ dimensions, a similar result holds Vempala, Wang:
  - $k = O(1)$ spherical gaussian components of S.D 1 each. Inter-center separation of at least a constant*.
  - We can correctly cluster EACH DATA POINT.

Inter-Center Distance and Clustering

- In 1-dimension, if we have two Gaussians of standard deviation (S.D.) 1 each:
  - If inter-center separation is at least 100 S.D.'s, we can tell them apart.
  - If inter-center separation is less than $1/100$, difficult to tell them apart.
- In $d-$ dimensions, a similar result holds Vempala, Wang:
  - $k = O(1)$ spherical gaussian components of S.D 1 each. Inter-center separation of at least a constant*.
  - We can correctly cluster EACH DATA POINT.
  - Use Singular Value Decomposition)(PCA) crucially. An elegant argument.

Theory explanation of practitioners' happiness

- Practitioners often solve NP-hard problems with heuristics and seem quite happy. Why? Perhaps, when the solution is **stable**, it is easy to find.

Theory explanation of practitioners' happiness

- Practitioners often solve NP-hard problems with heuristics and seem quite happy. Why? Perhaps, when the solution is **stable**, it is easy to find.
- Bilu, Lineal: Optimal solution to Max-cut is stable if arbitrary changes in edge weights each by factor $\leq \Delta$, leaves solution optimal. Q: For what values of Δ can we find stable solutions? Known only for $\Delta \geq \sqrt{n}$

Theory explanation of practitioners' happiness

- Practitioners often solve NP-hard problems with heuristics and seem quite happy. Why? Perhaps, when the solution is **stable**, it is easy to find.
- Bilu, Lineal: Optimal solution to Max-cut is stable if arbitrary changes in edge weights each by factor $\leq \Delta$, leaves solution optimal. Q: For what values of Δ can we find stable solutions? Known only for $\Delta \geq \sqrt{n}$
- Balcan, Blum, Gupta Opt $k-$means clustering C^* is stable if any near optimal clustering differs from C^* in a small fraction of objects. Algorithms to find stable solutions.Daniely, Lineal, Saks

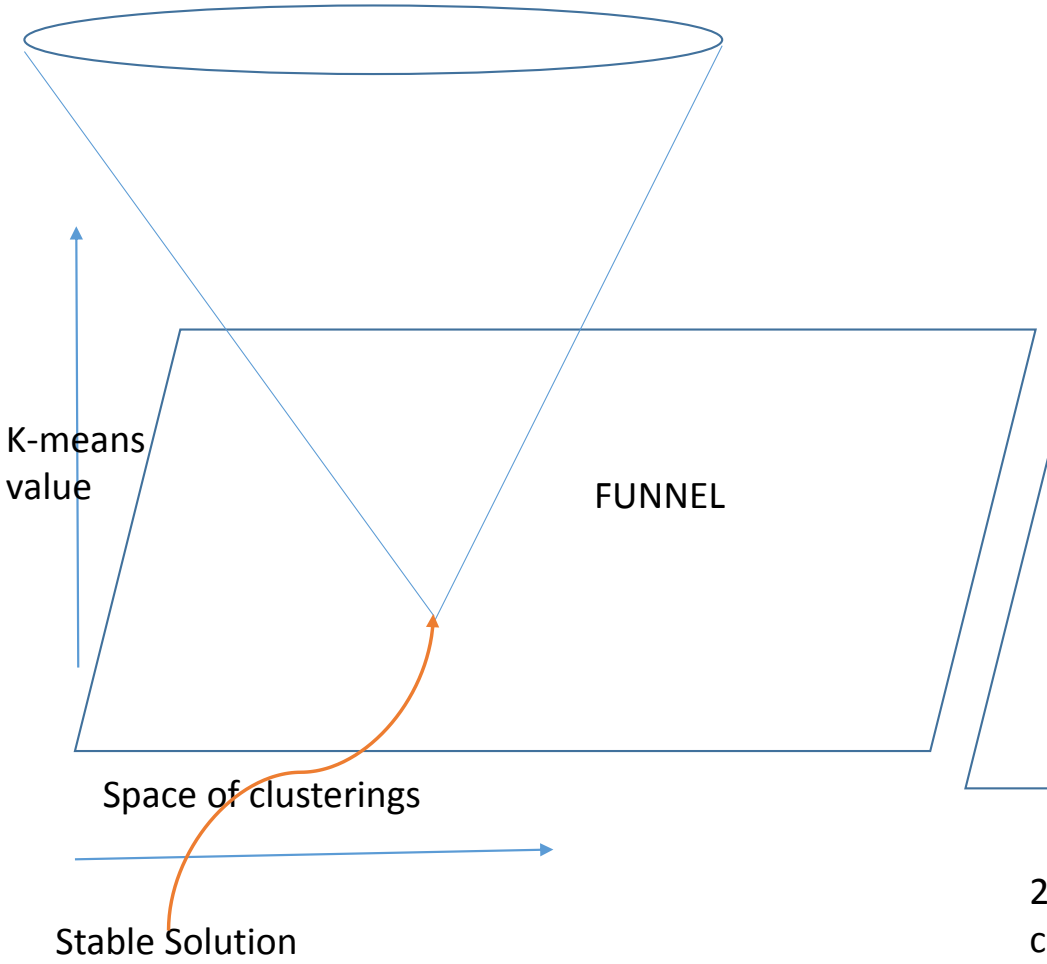
Theory explanation of practitioners' happiness

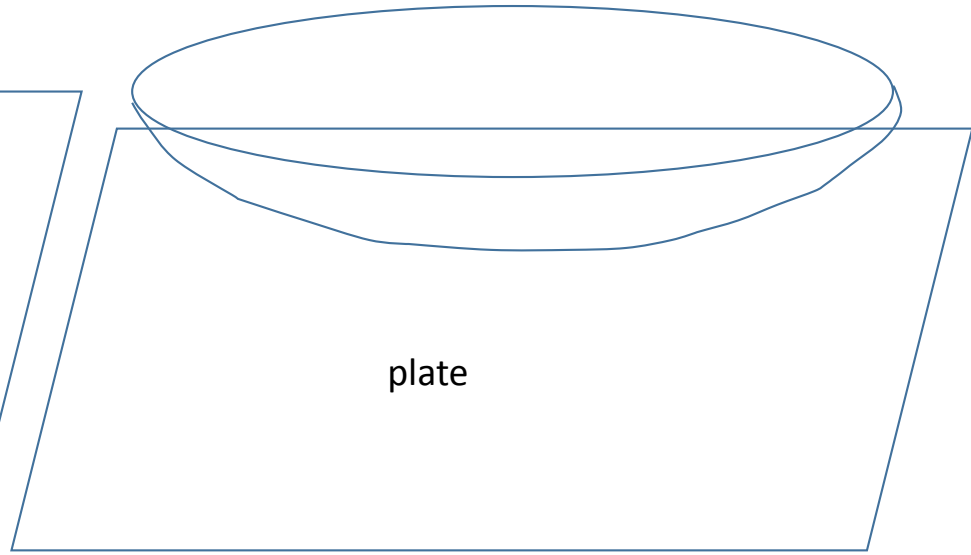
- Practitioners often solve NP-hard problems with heuristics and seem quite happy. Why? Perhaps, when the solution is **stable**, it is easy to find.
- Bilu, Lineal: Optimal solution to Max-cut is stable if arbitrary changes in edge weights each by factor $\leq \Delta$, leaves solution optimal. Q: For what values of Δ can we find stable solutions? Known only for $\Delta \geq \sqrt{n}$
- Balcan, Blum, Gupta Opt $k-$means clustering C^* is stable if any near optimal clustering differs from C^* in a small fraction of objects. Algorithms to find stable solutions.Daniely, Lineal, Saks
- Related Definitions: Balcan, Blum, Vempala; Ostravsky, Rabani, Schulman, Swamy; Awasthi, A. Blum, Sheffet : Optimal solution for $k-$ means is stable if it remains optimal even when we change pairwise distances, each by at most a constant factor.

Theory explanation of practitioners' happiness

- Practitioners often solve NP-hard problems with heuristics and seem quite happy. Why? Perhaps, when the solution is **stable**, it is easy to find.
- Bilu, Lineal: Optimal solution to Max-cut is stable if arbitrary changes in edge weights each by factor $\leq \Delta$, leaves solution optimal. Q: For what values of Δ can we find stable solutions? Known only for $\Delta \geq \sqrt{n}$
- Balcan, Blum, Gupta Opt $k-$means clustering C^* is stable if any near optimal clustering differs from C^* in a small fraction of objects. Algorithms to find stable solutions.Daniely, Lineal, Saks
- Related Definitions: Balcan, Blum, Vempala; Ostravsky, Rabani, Schulman, Swamy; Awasthi, A. Blum, Sheffet : Optimal solution for $k-$ means is stable if it remains optimal even when we change pairwise distances, each by at most a constant factor.
- Promising approaches and many open questions. But for the notorious two Gaussian picture, the correct solution is not stable!

Stability in Pictures





2-Gaussians. K-means value so large that mis-clustering points costs (relatively) little. So many clusterings of roughly the same value.

A new notion of stable clustering

- A $k-$clustering is proper

A new notion of stable clustering

- A $k-$clustering is proper
  - if its "variance" σ^2 (Max. over all directions u of the Mean Squared Distance of data points to their cluster centers in the direction u) is least among all k clusterings. (Optimize σ)

A new notion of stable clustering

- A $k-$clustering is proper
  - if its "variance" σ^2 (Max. over all directions u of the Mean Squared Distance of data points to their cluster centers in the direction u) is least among all k clusterings. (Optimize σ)
  - and inter-cluster-center separation is at least $c\sigma$. ["Means are 6 S.D.'s apart".]

A new notion of stable clustering

- A $k-$clustering is proper
  - if its "variance" σ^2 (Max. over all directions u of the Mean Squared Distance of data points to their cluster centers in the direction u) is least among all k clusterings. (Optimize σ)
  - and inter-cluster-center separation is at least $c\sigma$. ["Means are 6 S.D.'s apart".]
- Works for the two Gaussian picture.

A new notion of stable clustering

- A $k-$clustering is proper
  - if its "variance" σ^2 (Max. over all directions u of the Mean Squared Distance of data points to their cluster centers in the direction u) is least among all k clusterings. (Optimize σ)
  - and inter-cluster-center separation is at least $c\sigma$. ["Means are 6 S.D.'s apart".]
- Works for the two Gaussian picture.
- Hopcroft, Kannan If there is a proper clustering C^*, then Project and Cluster finds a clustering which has at most εn points classified differently than C^*.

A new notion of stable clustering

- A $k-$clustering is proper
  - if its "variance" σ^2 (Max. over all directions u of the Mean Squared Distance of data points to their cluster centers in the direction u) is least among all k clusterings. (Optimize σ)
  - and inter-cluster-center separation is at least $c\sigma$. ["Means are 6 S.D.'s apart".]
- Works for the two Gaussian picture.
- Hopcroft, Kannan If there is a proper clustering C^*, then Project and Cluster finds a clustering which has at most εn points classified differently than C^*.
- Mixture model $\rightarrow$ proper clustering.

A new notion of stable clustering

- A $k-$clustering is proper
  - if its "variance" σ^2 (Max. over all directions u of the Mean Squared Distance of data points to their cluster centers in the direction u) is least among all k clusterings. (Optimize σ)
  - and inter-cluster-center separation is at least $c\sigma$. ["Means are 6 S.D.'s apart".]
- Works for the two Gaussian picture.
- Hopcroft, Kannan If there is a proper clustering C^*, then Project and Cluster finds a clustering which has at most εn points classified differently than C^*.
- Mixture model $\rightarrow$ proper clustering.
- Harder Theorem: Kumar, Kannan: If there is a clustering C^* such that when data points are projected to space of centers, each projected data point is closer to its own center than any other center by at least $\Omega(\sigma(C^*))$, then Project and Cluster followed by Llyod's algorithm converges exponentially fast to the centers of C^*.

Lloyd's Algorithm

- Problem: Cluster data points in d space into k clusters.

Lloyd's Algorithm

- Problem: Cluster data points in d space into k clusters.
- **Algorithm**

Lloyd's Algorithm

- Problem: Cluster data points in d space into k clusters.
- **Algorithm**
  - Start with some k points as current cluster centers.

Lloyd's Algorithm

- Problem: Cluster data points in d space into k clusters.
- **Algorithm**
  - Start with some k points as current cluster centers.
  - Partition data points into k clusters based on nearest cluster center.

Lloyd's Algorithm

- Problem: Cluster data points in d space into k clusters.
- **Algorithm**
  - Start with some k points as current cluster centers.
  - Partition data points into k clusters based on nearest cluster center.
  - Recompute cluster centers as centroids of new clusters.

Lloyd's Algorithm

- Problem: Cluster data points in d space into k clusters.
- **Algorithm**
  - Start with some k points as current cluster centers.
  - Partition data points into k clusters based on nearest cluster center.
  - Recompute cluster centers as centroids of new clusters.
  - Repeat...

Lloyd's Algorithm

- Problem: Cluster data points in d space into k clusters.
- **Algorithm**
  - Start with some k points as current cluster centers.
  - Partition data points into k clusters based on nearest cluster center.
  - Recompute cluster centers as centroids of new clusters.
  - Repeat...
- Millions of happy users in ML,....

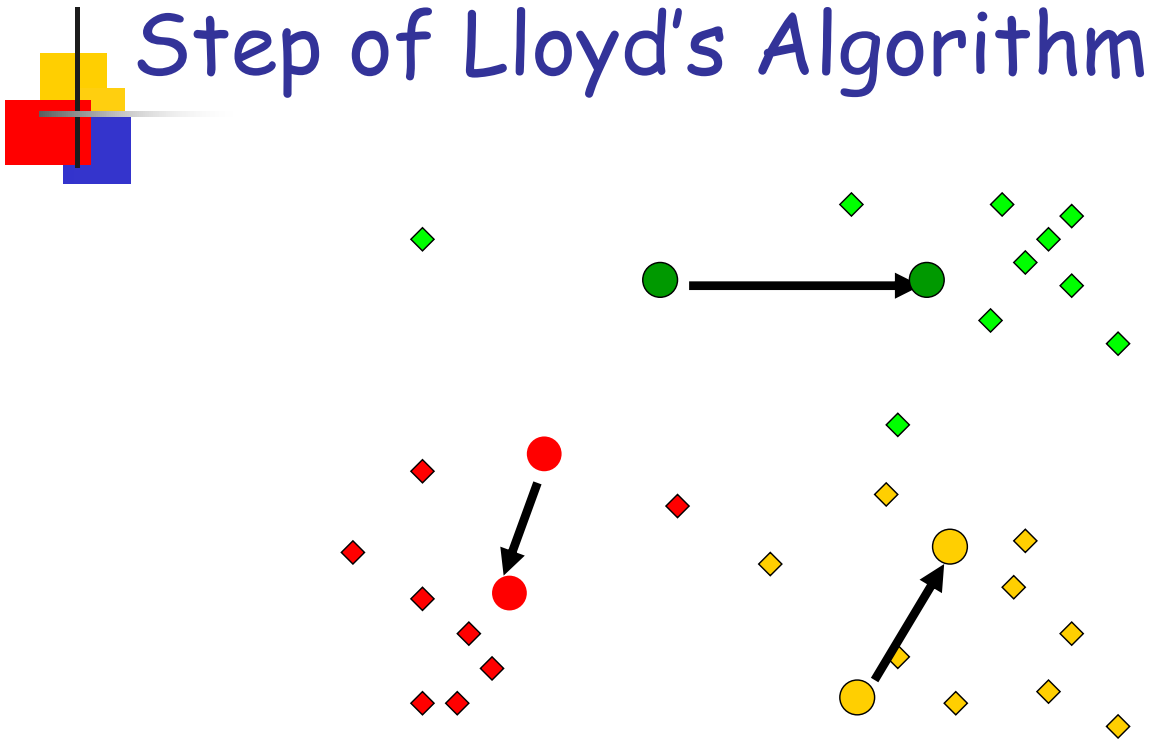
Lloyd's Algorithm

- Problem: Cluster data points in d space into k clusters.
- **Algorithm**
  - Start with some k points as current cluster centers.
  - Partition data points into k clusters based on nearest cluster center.
  - Recompute cluster centers as centroids of new clusters.
  - Repeat...
- Millions of happy users in ML,....
- Few unhappy theoreticians (cannot prove a lot)

Lloyd's Algorithm

- Problem: Cluster data points in d space into k clusters.
- **Algorithm**
  - Start with some k points as current cluster centers.
  - Partition data points into k clusters based on nearest cluster center.
  - Recompute cluster centers as centroids of new clusters.
  - Repeat...
- Millions of happy users in ML,....
- Few unhappy theoreticians (cannot prove a lot)
- Mean Squared Distance of data point to its cluster center = MSD (aka: k-means)
  - Lloyd's improves MSD at each step (SIMPLE).
  - Thus converges (to something).

Lloyd's Algorithm-Pictures

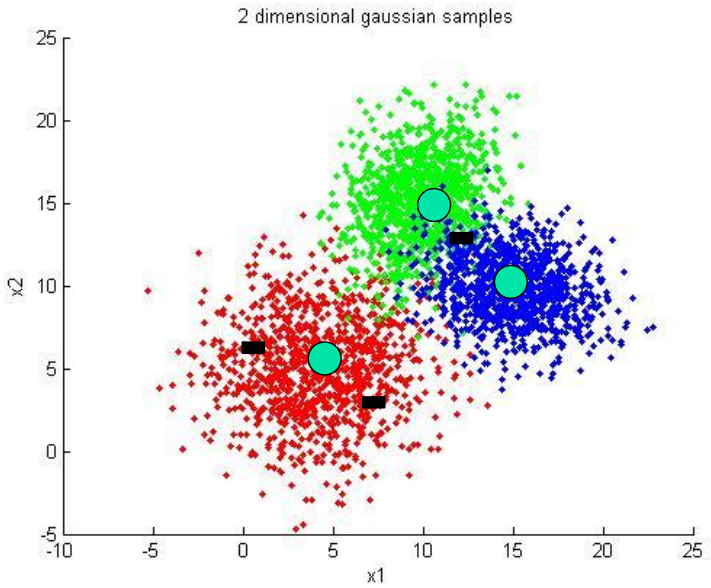


41

Pictures-II

Importance of a good start

- Good start
- Bad Start



More General Gaussians

Algorithm for fitting a general Mixture of k Gaussians

- Start with a initial guess of k (general) Gaussians: $(\mu_1, \Sigma_1), (\mu_2, \Sigma_2), \ldots, (\mu_k, \Sigma_k)$.

Any Provable Analysis (Besides just convergence to local opt. which follows from monotonicity).

More General Gaussians

Algorithm for fitting a general Mixture of k Gaussians

- Start with a initial guess of k (general) Gaussians: $(\mu_1, \Sigma_1), (\mu_2, \Sigma_2), \ldots, (\mu_k, \Sigma_k)$.
- For $t = 1, 2, \ldots, k$, make all data points whose (posterior) prob. according to (μ_t, Σ_t) is highest into cluster C_t.

Any Provable Analysis (Besides just convergence to local opt. which follows from monotonicity).

More General Gaussians

Algorithm for fitting a general Mixture of k Gaussians

- Start with a initial guess of k (general) Gaussians: $(\mu_1, \Sigma_1), (\mu_2, \Sigma_2), \ldots, (\mu_k, \Sigma_k)$.
- For $t = 1, 2, \ldots, k$, make all data points whose (posterior) prob. according to (μ_t, Σ_t) is highest into cluster C_t.
- Reset (μ_t, Σ_t) to be the sample mean and covarince of C_t.

Any Provable Analysis (Besides just convergence to local opt. which follows from monotonicity).

More General Gaussians

Algorithm for fitting a general Mixture of k Gaussians

- Start with a initial guess of k (general) Gaussians: $(\mu_1, \Sigma_1), (\mu_2, \Sigma_2), \ldots, (\mu_k, \Sigma_k)$.
- For $t = 1, 2, \ldots, k$, make all data points whose (posterior) prob. according to (μ_t, Σ_t) is highest into cluster C_t.
- Reset (μ_t, Σ_t) to be the sample mean and covarince of C_t.
- Repeat to heart's content.

Any Provable Analysis (Besides just convergence to local opt. which follows from monotonicity).

Nearest Neighbor Search

- Database of n points in $d-$dimensions. Preprocess at will.

Nearest Neighbor Search

- Database of n points in $d-$dimensions. Preprocess at will.
- Queries which are points in $d-$dim's will then arrive. Must quickly (logarithmic time) report the nearest (or approximately nearest) database point.

Nearest Neighbor Search

- Database of n points in $d-$dimensions. Preprocess at will.
- Queries which are points in $d-$dim's will then arrive. Must quickly (logarithmic time) report the nearest (or approximately nearest) database point.
- One of the most widely used subroutines. High d is a handicap.

Nearest Neighbor Search

- Database of n points in $d-$dimensions. Preprocess at will.
- Queries which are points in $d-$dim's will then arrive. Must quickly (logarithmic time) report the nearest (or approximately nearest) database point.
- One of the most widely used subroutines. High d is a handicap.
- Important tool in the theoretician's kit:
  Johnson Lindenstrauss Random Projection Theorem: v a fixed vector in $\mathbf{R}^d$. V a random k dimensional subspace of $\mathbf{R}^d$. Say v' is projection of v onto V. With high probability,
  $|v'| \approx \frac{\sqrt{k}}{\sqrt{d}}|v|$.

Nearest Neighbor Search

- Database of n points in $d-$dimensions. Preprocess at will.
- Queries which are points in $d-$dim's will then arrive. Must quickly (logarithmic time) report the nearest (or approximately nearest) database point.
- One of the most widely used subroutines. High d is a handicap.
- Important tool in the theoretician's kit:
  Johnson Lindenstrauss Random Projection Theorem: v a fixed vector in $\mathbf{R}^d$. V a random k dimensional subspace of $\mathbf{R}^d$. Say v' is projection of v onto V. With high probability,
  $|v'| \approx \frac{\sqrt{k}}{\sqrt{d}}|v|$.
- Failure Probability is low; so can preserve all pairwise distances among n points in $\mathbf{R}^d$ with k only about $\ln n$.

Nearest Neighbor Search

- Database of n points in $d-$dimensions. Preprocess at will.
- Queries which are points in $d-$dim's will then arrive. Must quickly (logarithmic time) report the nearest (or approximately nearest) database point.
- One of the most widely used subroutines. High d is a handicap.
- Important tool in the theoretician's kit:
  Johnson Lindenstrauss Random Projection Theorem: v a fixed vector in $\mathbf{R}^d$. V a random k dimensional subspace of $\mathbf{R}^d$. Say v' is projection of v onto V. With high probability,
  $|v'| \approx \frac{\sqrt{k}}{\sqrt{d}}|v|$.
- Failure Probability is low; so can preserve all pairwise distances among n points in $\mathbf{R}^d$ with k only about $\ln n$.
- Kleinberg; Indyk, Motwani Project Data points into a random low dimensional subspace and find NN to query point in the projection.

Tale of two Dimension Reduction Methods

- PCA projects high dim'l data to best-fit subspace. Used in practice a lot. Clustering is one area where we have proofs of its efficacy. Rather rare.

Tale of two Dimension Reduction Methods

- PCA projects high dim'l data to best-fit subspace. Used in practice a lot. Clustering is one area where we have proofs of its efficacy. Rather rare.
- Random Projections used widely in theory with provable efficacy. But for NNS, PCA is used as well in practice.

Tale of two Dimension Reduction Methods

- PCA projects high dim'l data to best-fit subspace. Used in practice a lot. Clustering is one area where we have proofs of its efficacy. Rather rare.
- Random Projections used widely in theory with provable efficacy. But for NNS, PCA is used as well in practice.
- Prove that PCA does the job in NNS and other applications.

Tale of two Dimension Reduction Methods

- PCA projects high dim'l data to best-fit subspace. Used in practice a lot. Clustering is one area where we have proofs of its efficacy. Rather rare.
- Random Projections used widely in theory with provable efficacy. But for NNS, PCA is used as well in practice.
- Prove that PCA does the job in NNS and other applications.
- Difficulty: Whereas Random Projections preserve EVERY (pairwise) distance, PCA does not. But surely, data independent random projection just cannot always be that good?

The correct clustering: Abstract View

- Kleinberg: A Clustering criterion Γ (such as $k-$means) gives a mapping
  Distance Function $d(x, y) \rightarrow$ (Optimal) Partition $\Gamma(d)$ of data points.

The correct clustering: Abstract View

- Kleinberg: A Clustering criterion Γ (such as $k-$means) gives a mapping
  Distance Function $d(x, y) \rightarrow$ (Optimal) Partition $\Gamma(d)$ of data points.
- A System of "reasonable" axioms any clustering criterion ought to satisfy:

The correct clustering: Abstract View

- Kleinberg: A Clustering criterion Γ (such as $k-$means) gives a mapping
  Distance Function $d(x, y) \rightarrow$ (Optimal) Partition $\Gamma(d)$ of data points.
- A System of "reasonable" axioms any clustering criterion ought to satisfy:
  - Consistency: If we increase distances between points in different clusters and decrease distances between points in same cluster, the optimal clustering should still remain optimal. (Beware: Ties)

The correct clustering: Abstract View

- Kleinberg: A Clustering criterion Γ (such as $k-$means) gives a mapping
  Distance Function $d(x, y) \rightarrow$ (Optimal) Partition $\Gamma(d)$ of data points.
- A System of "reasonable" axioms any clustering criterion ought to satisfy:
  - Consistency: If we increase distances between points in different clusters and decrease distances between points in same cluster, the optimal clustering should still remain optimal. (Beware: Ties)
  - Scale Invariance Multiplying all distances by the same constant leaves the optimal clustering still optimal.

The correct clustering: Abstract View

- Kleinberg: A Clustering criterion Γ (such as $k-$means) gives a mapping
  Distance Function $d(x, y) \rightarrow$ (Optimal) Partition $\Gamma(d)$ of data points.
- A System of "reasonable" axioms any clustering criterion ought to satisfy:
  - Consistency: If we increase distances between points in different clusters and decrease distances between points in same cluster, the optimal clustering should still remain optimal. (Beware: Ties)
  - Scale Invariance Multiplying all distances by the same constant leaves the optimal clustering still optimal.
  - Richness For any partition $\mathcal{P}$ of n data points, there is some distance function $d(x, y)$ on the points for which $\Gamma(d) = \mathcal{P}$.

The correct clustering: Abstract View

- Kleinberg: A Clustering criterion Γ (such as $k-$means) gives a mapping
  Distance Function $d(x, y) \rightarrow$ (Optimal) Partition $\Gamma(d)$ of data points.
- A System of "reasonable" axioms any clustering criterion ought to satisfy:
  - Consistency: If we increase distances between points in different clusters and decrease distances between points in same cluster, the optimal clustering should still remain optimal. (Beware: Ties)
  - Scale Invariance Multiplying all distances by the same constant leaves the optimal clustering still optimal.
  - Richness For any partition $\mathcal{P}$ of n data points, there is some distance function $d(x, y)$ on the points for which $\Gamma(d) = \mathcal{P}$.
- Theorem There is no clustering criterion satisfying all the axioms.

The correct clustering: Abstract View

- Kleinberg: A Clustering criterion Γ (such as $k-$means) gives a mapping
  Distance Function $d(x, y) \rightarrow$ (Optimal) Partition $\Gamma(d)$ of data points.
- A System of "reasonable" axioms any clustering criterion ought to satisfy:
  - Consistency: If we increase distances between points in different clusters and decrease distances between points in same cluster, the optimal clustering should still remain optimal. (Beware: Ties)
  - Scale Invariance Multiplying all distances by the same constant leaves the optimal clustering still optimal.
  - Richness For any partition $\mathcal{P}$ of n data points, there is some distance function $d(x, y)$ on the points for which $\Gamma(d) = \mathcal{P}$.
- Theorem There is no clustering criterion satisfying all the axioms.
- Oops???

A Feasible set of Axioms

- If we insist on points being in Euclidean space, there is a criterion satisfying the axioms:

A Feasible set of Axioms

- If we insist on points being in Euclidean space, there is a criterion satisfying the axioms:
  - Consistency: If we move a point so that its distance to points in its cluster decreases and to points in different clusters increases, the optimal clustering should still remain optimal. (Beware: Ties)

A Feasible set of Axioms

- If we insist on points being in Euclidean space, there is a criterion satisfying the axioms:
  - Consistency: If we move a point so that its distance to points in its cluster decreases and to points in different clusters increases, the optimal clustering should still remain optimal. (Beware: Ties)

A Feasible set of Axioms

- If we insist on points being in Euclidean space, there is a criterion satisfying the axioms:
  - Consistency: If we move a point so that its distance to points in its cluster decreases and to points in different clusters increases, the optimal clustering should still remain optimal. (Beware: Ties)
  - Scale Invariance Multiplying all distances by the same constant leaves the optimal clustering still optimal.

A Feasible set of Axioms

- If we insist on points being in Euclidean space, there is a criterion satisfying the axioms:
  - Consistency: If we move a point so that its distance to points in its cluster decreases and to points in different clusters increases, the optimal clustering should still remain optimal. (Beware: Ties)
  - Scale Invariance Multiplying all distances by the same constant leaves the optimal clustering still optimal.
  - Richness For any set K of k points in space, there is some placement of the n data points so that the clustering with K as centers is optimal.

A Feasible set of Axioms

- If we insist on points being in Euclidean space, there is a criterion satisfying the axioms:
  - Consistency: If we move a point so that its distance to points in its cluster decreases and to points in different clusters increases, the optimal clustering should still remain optimal. (Beware: Ties)
  - Scale Invariance Multiplying all distances by the same constant leaves the optimal clustering still optimal.
  - Richness For any set K of k points in space, there is some placement of the n data points so that the clustering with K as centers is optimal.
- Hopcroft, Kannan: Theorem: Balanced $k-$means (Minimum sum of dist squared to cluster centers among all partitions into k clusters, each of size n/k) satisfies all the axioms.