Fast low-rank metric learning

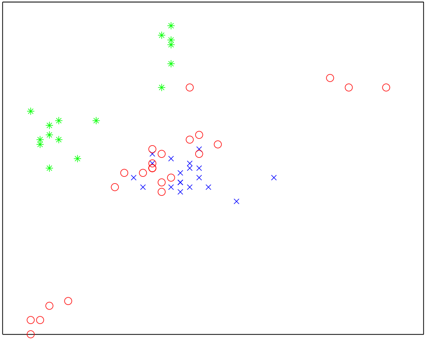
Dan-Theodor Oneaţă

July 25, 2011

k nearest neighbours

- Simple, yet powerful classifier.
- Euclidean distance:

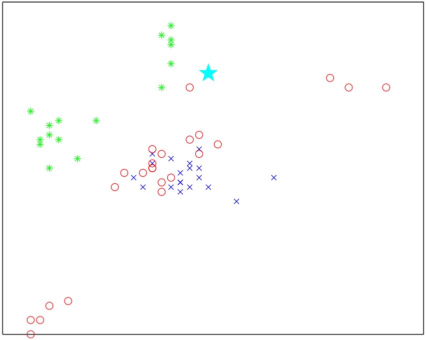
$$d(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{(\mathbf{x}_i - \mathbf{x}_j)^{\mathrm{T}}(\mathbf{x}_i - \mathbf{x}_j)}$$



k nearest neighbours

- Simple, yet powerful classifier.
- Euclidean distance:

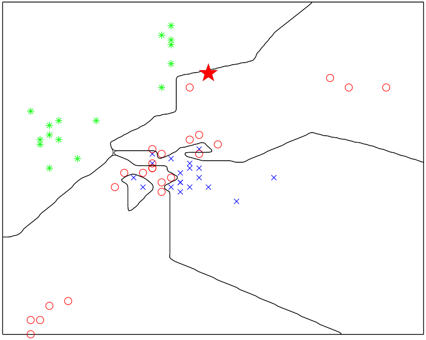
$$d(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{(\mathbf{x}_i - \mathbf{x}_j)^{\mathrm{T}}(\mathbf{x}_i - \mathbf{x}_j)}$$



k nearest neighbours

- Simple, yet powerful classifier.
- Euclidean distance:

$$d(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{(\mathbf{x}_i - \mathbf{x}_j)^{\mathrm{T}}(\mathbf{x}_i - \mathbf{x}_j)}$$

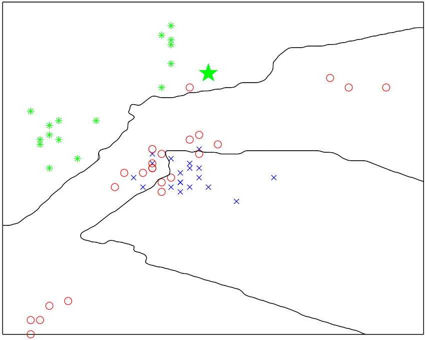


Decision boundaries for $k = 1$

k nearest neighbours

- Simple, yet powerful classifier.
- Euclidean distance:

$$d(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{(\mathbf{x}_i - \mathbf{x}_j)^{\mathrm{T}}(\mathbf{x}_i - \mathbf{x}_j)}$$

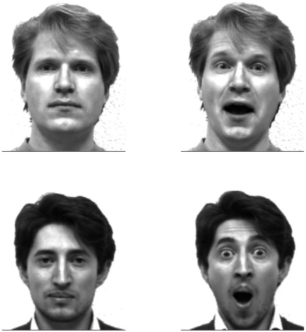


Decision boundaries for $k = 7$

k nearest neighbours

- Simple, yet powerful classifier.
- Euclidean distance:

$$d(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{(\mathbf{x}_i - \mathbf{x}_j)^{\mathrm{T}}(\mathbf{x}_i - \mathbf{x}_j)}$$



k nearest neighbours

- Simple, yet powerful classifier.
- Euclidean distance:

$$d(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{(\mathbf{x}_i - \mathbf{x}_j)^{\mathrm{T}}(\mathbf{x}_i - \mathbf{x}_j)}$$

Face recognition

k nearest neighbours

- Simple, yet powerful classifier.
- Euclidean distance:

$$d(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{(\mathbf{x}_i - \mathbf{x}_j)^{\mathrm{T}}(\mathbf{x}_i - \mathbf{x}_j)}$$

Expression recognition

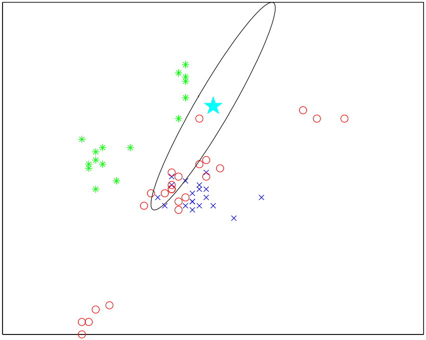
Neighbourhood component analysis

(Goldberger et at, 2004)

- Learns a Mahalanobis metric

  $$d_{\mathbf{S}}(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{(\mathbf{x}_i - \mathbf{x}_j)^{\mathrm{T}} \mathbf{S} (\mathbf{x}_i - \mathbf{x}_j)}$$

- Equivalent to a linear transformation:
  $d_{\mathbf{S}}(\mathbf{x}_i, \mathbf{x}_j) = d_{\mathbf{I}}(\mathbf{A}\mathbf{x}_i, \mathbf{A}\mathbf{x}_j)$



Mahalanobis metric

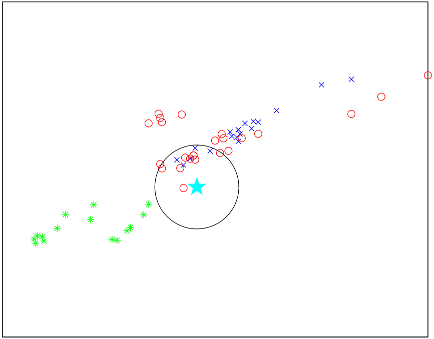
Neighbourhood component analysis

(Goldberger et at, 2004)

- Learns a Mahalanobis metric

$$d_{\mathbf{S}}(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{(\mathbf{x}_i - \mathbf{x}_j)^{\mathrm{T}}\mathbf{S}(\mathbf{x}_i - \mathbf{x}_j)}$$

- Equivalent to a linear transformation:
$d_{\mathbf{S}}(\mathbf{x}_i, \mathbf{x}_j) = d_{\mathbf{I}}(\mathbf{A}\mathbf{x}_i, \mathbf{A}\mathbf{x}_j)$



Euclidean metric in $\mathbf{AX}$

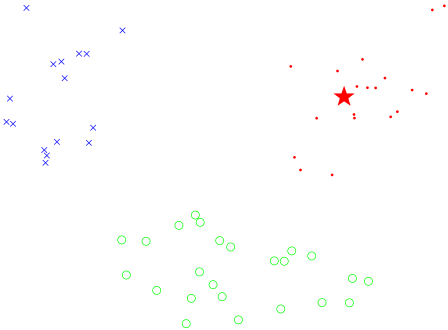
Neighbourhood component analysis

(Goldberger et at, 2004)

1. Find $\mathbf{A}$ that maximizes leave-one-out cross-validation score.
2. Soft version:

$$p(\mathbf{x}_i \in \text{class } c) = \frac{\sum_{j \in c} \exp\{-d^2_{\mathbf{S}}(\mathbf{x}_i, \mathbf{x}_j)\}}{\sum_k \exp\{-d^2_{\mathbf{S}}(\mathbf{x}_i, \mathbf{x}_k)\}}$$

$$\text{Maximize} f(\mathbf{A}) = \sum_i p(\mathbf{x}_i \in \text{true class of } \mathbf{x}_i).$$



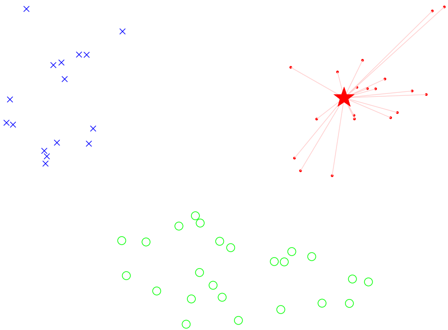
Neighbourhood component analysis

(Goldberger et at, 2004)

1. Find $\mathbf{A}$ that maximizes leave-one-out cross-validation score.
2. Soft version:

$$p(\mathbf{x}_i \in \text{class } c) = \frac{\sum_{j \in c} \exp\{-d^2_{\mathbf{S}}(\mathbf{x}_i, \mathbf{x}_j)\}}{\sum_k \exp\{-d^2_{\mathbf{S}}(\mathbf{x}_i, \mathbf{x}_k)\}}$$

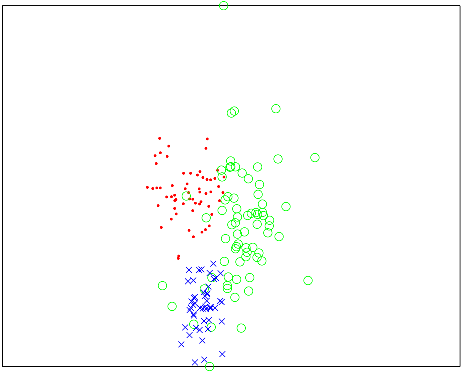
$$\text{Maximize} f(\mathbf{A}) = \sum_i p(\mathbf{x}_i \in \text{true class of } \mathbf{x}_i).$$



Optimizing $f(\mathbf{A})$

- Use $\nabla_{\mathbf{A}} f(\mathbf{A})$ for an optimization algorithm: *e.g.*, gradient ascent, conjugate gradients.
- How to initialise? Use random $\mathbf{A}$ or most discriminative projections given by PCA, LDA or logistic regression.

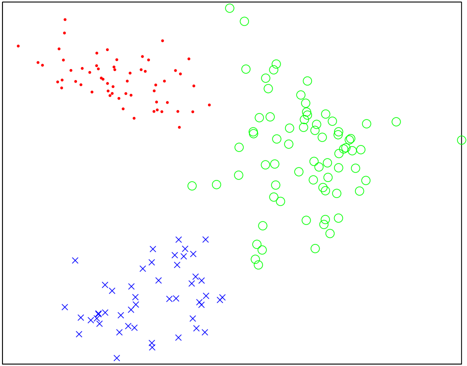
Initial random projection



`A=randn(d,D)`

Optimizing $f(\mathbf{A})$

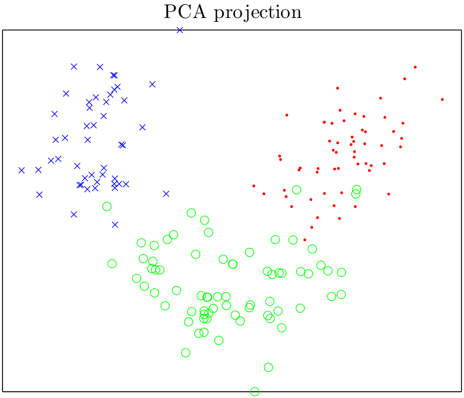
- Use $\nabla_{\mathbf{A}} f(\mathbf{A})$ for an optimization algorithm: *e.g.*, gradient ascent, conjugate gradients.
- How to initialise? Use random $\mathbf{A}$ or most discriminative projections given by PCA, LDA or logistic regression.



`A=maximize('nca',A)`

Optimizing $f(\mathbf{A})$

- Use $\nabla_{\mathbf{A}} f(\mathbf{A})$ for an optimization algorithm: *e.g.*, gradient ascent, conjugate gradients.
- How to initialise? Use random $\mathbf{A}$ or most discriminative projections given by PCA, LDA or logistic regression.

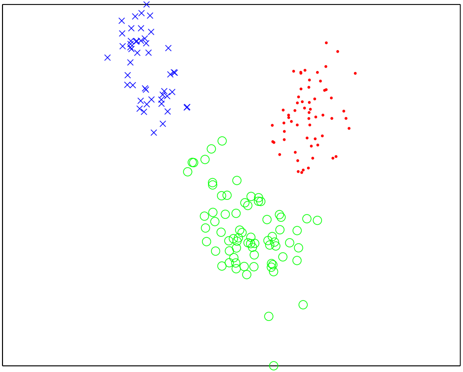


`A=eig(X*X'/N)`

Optimizing $f(\mathbf{A})$

- Use $\nabla_{\mathbf{A}} f(\mathbf{A})$ for an optimization algorithm: *e.g.*, gradient ascent, conjugate gradients.
- How to initialise? Use random $\mathbf{A}$ or most discriminative projections given by PCA, LDA or logistic regression.

Final projection

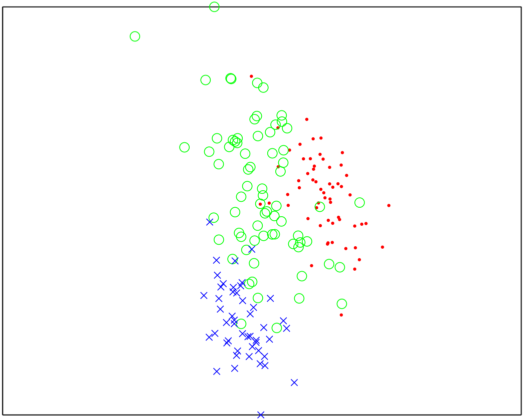


`A=maximize('nca',A)`

Speeding up the computations

1. Sub-sample the data set.
2. Use mini-batches:
   - Choose them randomly
   - Use cheap clustering method.

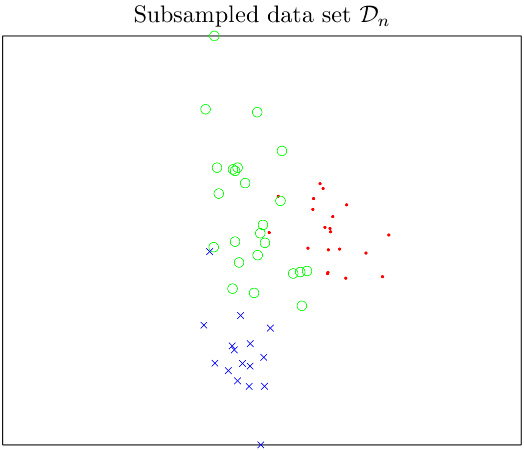
Initial projection of $\mathcal{D}$



Sub-sampling example

Speeding up the computations

1. Sub-sample the data set.
2. Use mini-batches:
   - Choose them randomly
   - Use cheap clustering method.

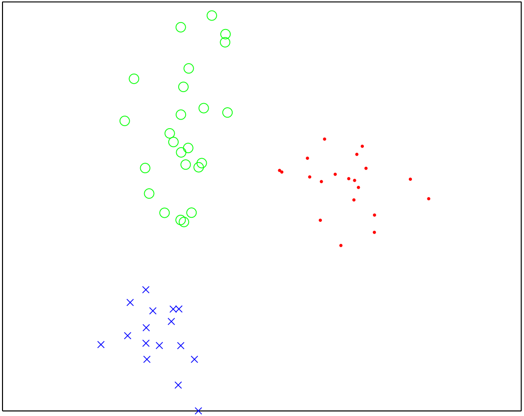


Sub-sampling example

Speeding up the computations

1. Sub-sample the data set.
2. Use mini-batches:
   - Choose them randomly
   - Use cheap clustering method.

Final projection on $\mathcal{D}_n$

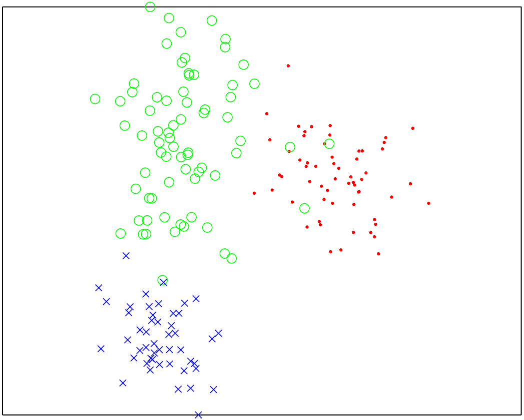


Sub-sampling example

Speeding up the computations

1. Sub-sample the data set.
2. Use mini-batches:
   - Choose them randomly
   - Use cheap clustering method.

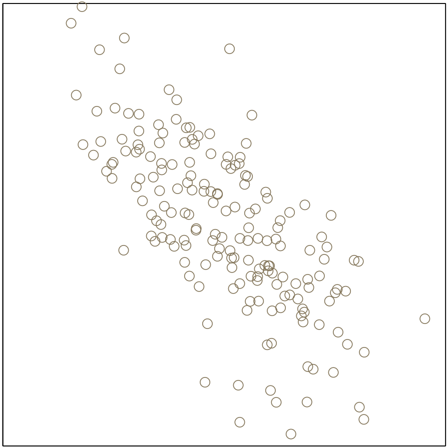
Final projection on $\mathcal{D}$



Sub-sampling example

Speeding up the computations

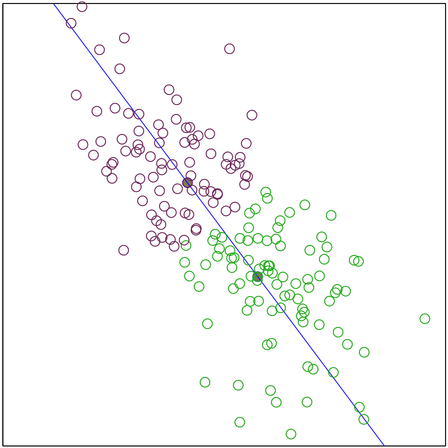
1. Sub-sample the data set.
2. Use mini-batches:
   - Choose them randomly
   - Use cheap clustering method.



Recursive projection clustering

Speeding up the computations

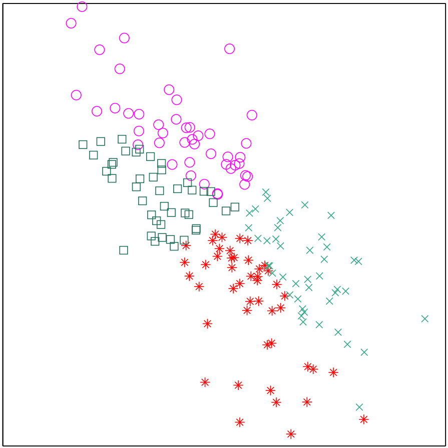
1. Sub-sample the data set.
2. Use mini-batches:
   - Choose them randomly
   - Use cheap clustering method.



Recursive projection clustering

Speeding up the computations

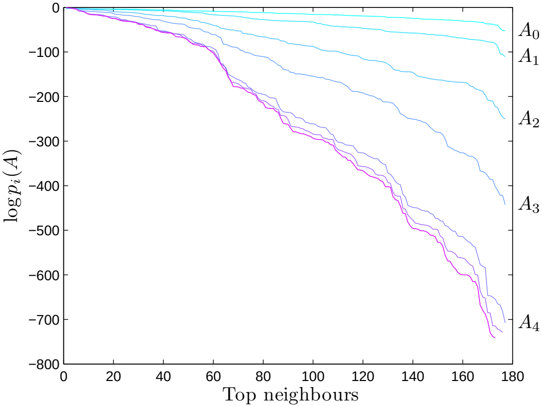
1. Sub-sample the data set.
2. Use mini-batches:
   - Choose them randomly
   - Use cheap clustering method.



Recursive projection clustering

Approximate computations

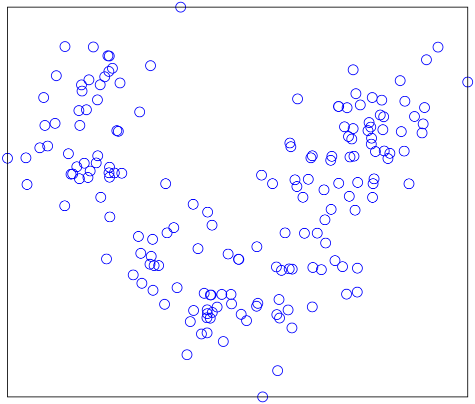
- Each contribution $p_{ij} \propto \exp\{-d^2(\mathbf{A}\mathbf{x}_i, \mathbf{A}\mathbf{x}_j)\}$.
- Cast NCA into class conditional kernel density estimation problem: $p(\mathbf{x}_i|c)$.
- Use k-d trees for fast density estimation.



How p_{ij} varies during training

Approximate computations

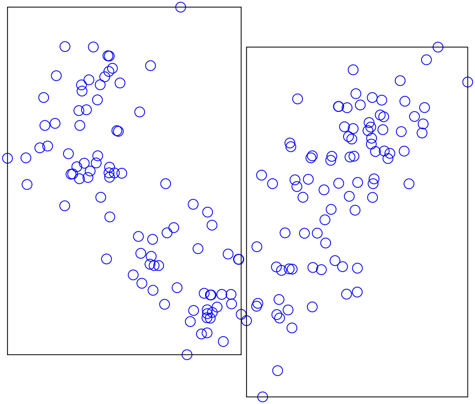
- Each contribution $p_{ij} \propto \exp\{-d^2(\mathbf{A}\mathbf{x}_i, \mathbf{A}\mathbf{x}_j)\}$.
- Cast NCA into class conditional kernel density estimation problem: $p(\mathbf{x}_i|c)$.
- Use k-d trees for fast density estimation.



k-d tree example

Approximate computations

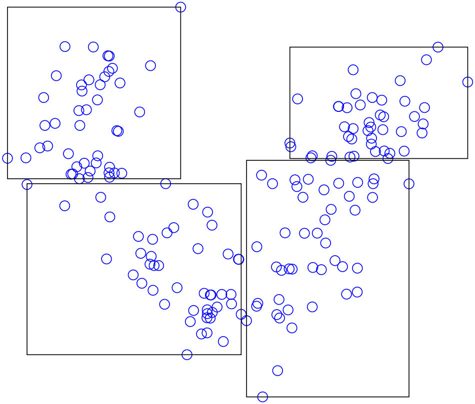
- Each contribution $p_{ij} \propto \exp\{-d^2(\mathbf{A}\mathbf{x}_i, \mathbf{A}\mathbf{x}_j)\}$.
- Cast NCA into class conditional kernel density estimation problem: $p(\mathbf{x}_i|c)$.
- Use k-d trees for fast density estimation.



k-d tree example

Approximate computations

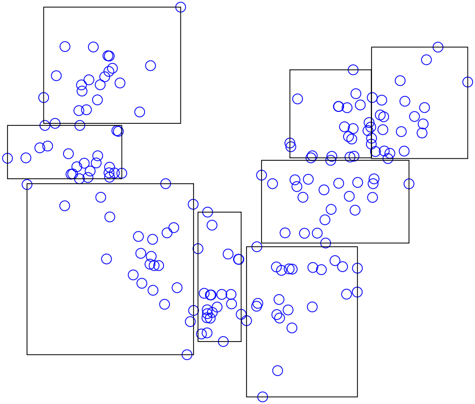
- Each contribution $p_{ij} \propto \exp\{-d^2(\mathbf{Ax}_i, \mathbf{Ax}_j)\}$.
- Cast NCA into class conditional kernel density estimation problem: $p(\mathbf{x}_i|c)$.
- Use k-d trees for fast density estimation.



k-d tree example

Approximate computations

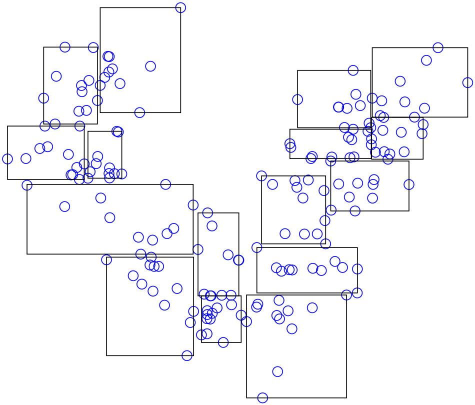
- Each contribution $p_{ij} \propto \exp\{-d^2(\mathbf{Ax}_i, \mathbf{Ax}_j)\}$.
- Cast NCA into class conditional kernel density estimation problem: $p(\mathbf{x}_i|c)$.
- Use k-d trees for fast density estimation.



k-d tree example

Approximate computations

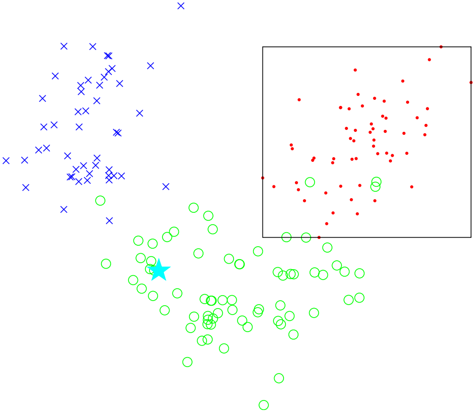
- Each contribution $p_{ij} \propto \exp\{-d^2(\mathbf{A}\mathbf{x}_i, \mathbf{A}\mathbf{x}_j)\}$.
- Cast NCA into class conditional kernel density estimation problem: $p(\mathbf{x}_i|c)$.
- Use k-d trees for fast density estimation.



k-d tree example

Approximate computations

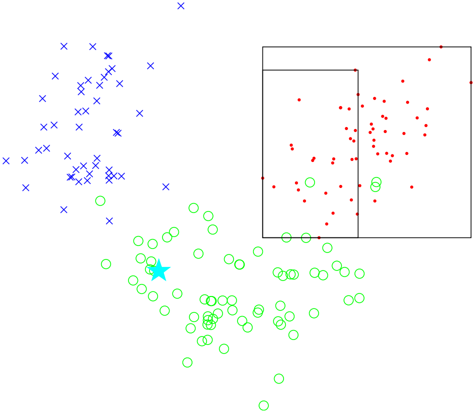
- Each contribution $p_{ij} \propto \exp\{-d^2(\mathbf{A}\mathbf{x}_i, \mathbf{A}\mathbf{x}_j)\}$.
- Cast NCA into class conditional kernel density estimation problem: $p(\mathbf{x}_i|c)$.
- Use k-d trees for fast density estimation.



CC-KDE example: $p(\mathbf{x}_i|c = \text{red})$

Approximate computations

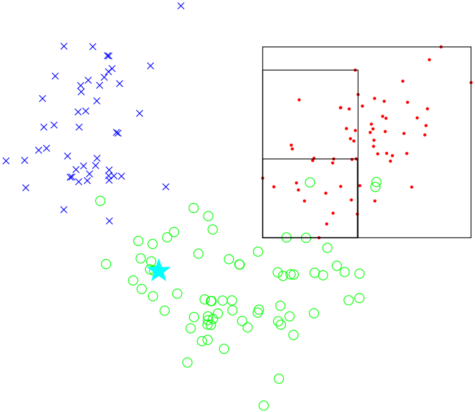
- Each contribution $p_{ij} \propto \exp\{-d^2(\mathbf{A}\mathbf{x}_i, \mathbf{A}\mathbf{x}_j)\}$.
- Cast NCA into class conditional kernel density estimation problem: $p(\mathbf{x}_i|c)$.
- Use k-d trees for fast density estimation.



CC-KDE example: $p(\mathbf{x}_i|c = \text{red})$

Approximate computations

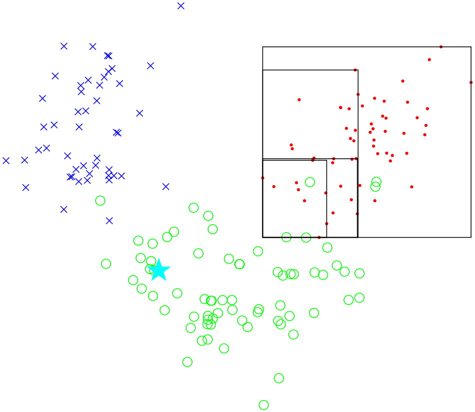
- Each contribution $p_{ij} \propto \exp\{-d^2(\mathbf{Ax}_i, \mathbf{Ax}_j)\}$.
- Cast NCA into class conditional kernel density estimation problem: $p(\mathbf{x}_i|c)$.
- Use k-d trees for fast density estimation.



CC-KDE example: $p(\mathbf{x}_i|c = \text{red})$

Approximate computations

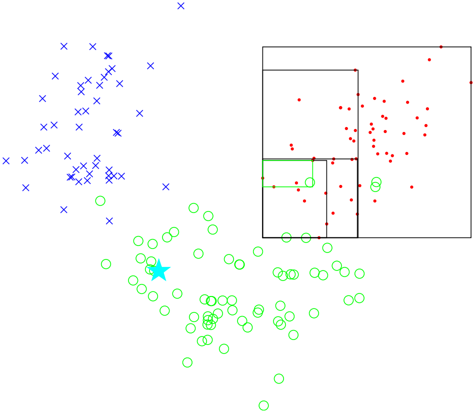
- Each contribution $p_{ij} \propto \exp\{-d^2(\mathbf{A}\mathbf{x}_i, \mathbf{A}\mathbf{x}_j)\}$.
- Cast NCA into class conditional kernel density estimation problem: $p(\mathbf{x}_i|c)$.
- Use k-d trees for fast density estimation.



CC-KDE example: $p(\mathbf{x}_i|c = \text{red})$

Approximate computations

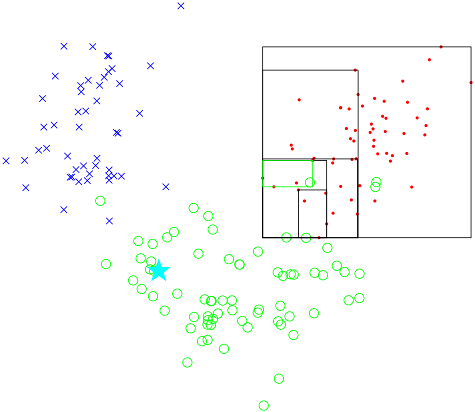
- Each contribution $p_{ij} \propto \exp\{-d^2(\mathbf{Ax}_i, \mathbf{Ax}_j)\}$.
- Cast NCA into class conditional kernel density estimation problem: $p(\mathbf{x}_i|c)$.
- Use k-d trees for fast density estimation.



CC-KDE example: $p(\mathbf{x}_i|c = \text{red})$

Approximate computations

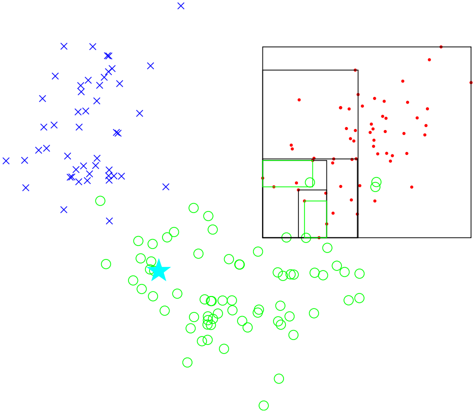
- Each contribution $p_{ij} \propto \exp\{-d^2(\mathbf{Ax}_i, \mathbf{Ax}_j)\}$.
- Cast NCA into class conditional kernel density estimation problem: $p(\mathbf{x}_i|c)$.
- Use k-d trees for fast density estimation.



CC-KDE example: $p(\mathbf{x}_i|c = \text{red})$

Approximate computations

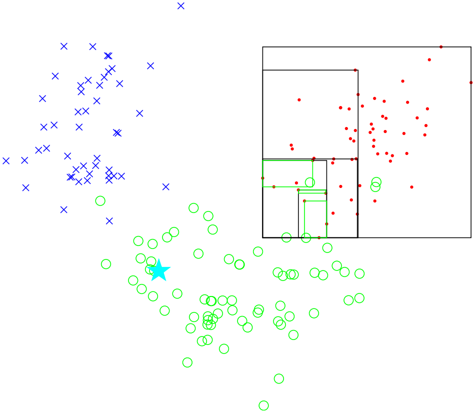
- Each contribution $p_{ij} \propto \exp\{-d^2(\mathbf{A}\mathbf{x}_i, \mathbf{A}\mathbf{x}_j)\}$.
- Cast NCA into class conditional kernel density estimation problem: $p(\mathbf{x}_i|c)$.
- Use k-d trees for fast density estimation.



CC-KDE example: $p(\mathbf{x}_i|c = \text{red})$

Approximate computations

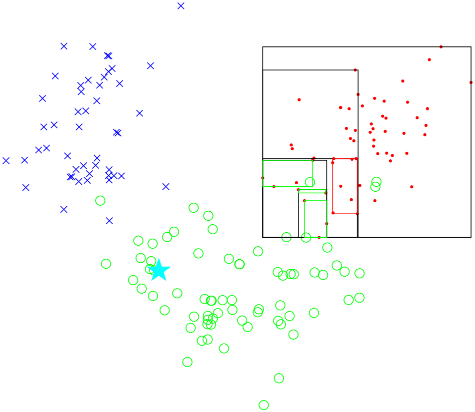
- Each contribution $p_{ij} \propto \exp\{-d^2(\mathbf{A}\mathbf{x}_i, \mathbf{A}\mathbf{x}_j)\}$.
- Cast NCA into class conditional kernel density estimation problem: $p(\mathbf{x}_i|c)$.
- Use k-d trees for fast density estimation.



CC-KDE example: $p(\mathbf{x}_i|c = \text{red})$

Approximate computations

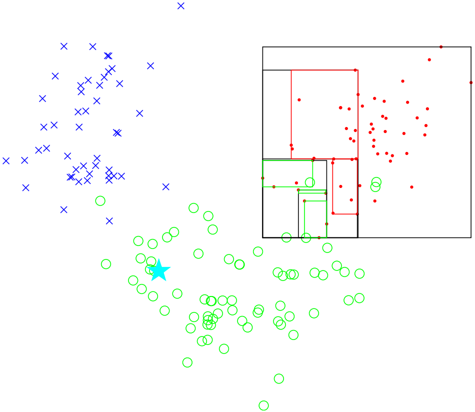
- Each contribution $p_{ij} \propto \exp\{-d^2(\mathbf{A}\mathbf{x}_i, \mathbf{A}\mathbf{x}_j)\}$.
- Cast NCA into class conditional kernel density estimation problem: $p(\mathbf{x}_i|c)$.
- Use k-d trees for fast density estimation.



CC-KDE example: $p(\mathbf{x}_i|c = \text{red})$

Approximate computations

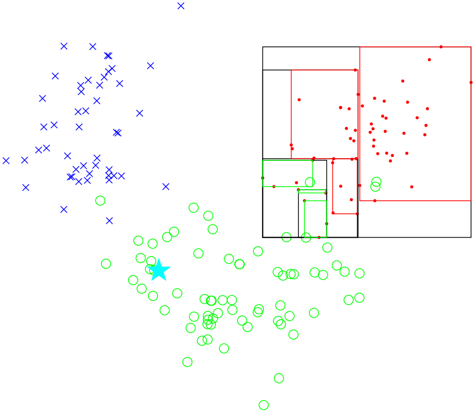
- Each contribution $p_{ij} \propto \exp\{-d^2(\mathbf{A}\mathbf{x}_i, \mathbf{A}\mathbf{x}_j)\}$.
- Cast NCA into class conditional kernel density estimation problem: $p(\mathbf{x}_i|c)$.
- Use k-d trees for fast density estimation.



CC-KDE example: $p(\mathbf{x}_i|c = \text{red})$

Approximate computations

- Each contribution $p_{ij} \propto \exp\{-d^2(\mathbf{A}\mathbf{x}_i, \mathbf{A}\mathbf{x}_j)\}$.
- Cast NCA into class conditional kernel density estimation problem: $p(\mathbf{x}_i|c)$.
- Use k-d trees for fast density estimation.



CC-KDE example: $p(\mathbf{x}_i|c = \text{red})$

Work done

- Introduction
- Background
  - Theoretical background
  - Related methods
- Neighbourhood component analysis
  - General presentation
  - Practical issues
  - Interpreting NCA as class-conditional density estimation problem
- Reducing computational cost
  - Sub-sampling
  - Mini-batches
  - Stochastic learning
  - Approximate computations
  - Exact computations using compact support kernels
  - NCA with compact support kernels and background distribution
- Evaluation (on large data sets)
- Conclusions

Remaining work

- Introduction
- Background
  - Theoretical background
  - Related methods
- Neighbourhood component analysis
  - General presentation
  - Practical issues
  - Interpreting NCA as class-conditional density estimation problem
- Reducing computational cost
  - Sub-sampling
  - Mini-batches
  - Stochastic learning
  - Approximate computations
  - Exact computations using compact support kernels
  - NCA with compact support kernels and background distribution
- Evaluation (on large data sets)
- Conclusions

Remaining work

- Introduction
- Background
  - Theoretical background
  - Related methods
- Neighbourhood component analysis
  - General presentation
  - Practical issues
  - Interpreting NCA as class-conditional density estimation problem
- Reducing computational cost
  - Sub-sampling
  - Mini-batches
  - Stochastic learning
  - Approximate computations
  - Exact computations using compact support kernels
  - NCA with compact support kernels and background distribution
- Evaluation (on large data sets)
- Conclusions

THANK YOU!