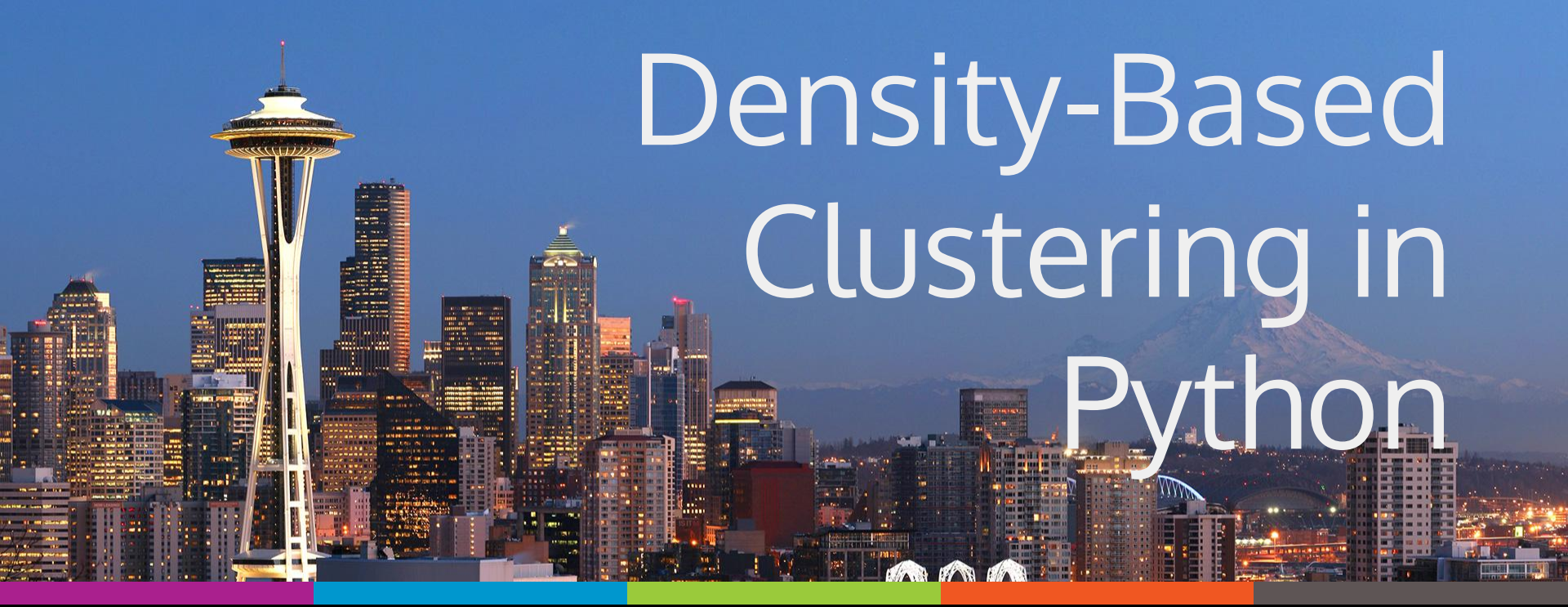


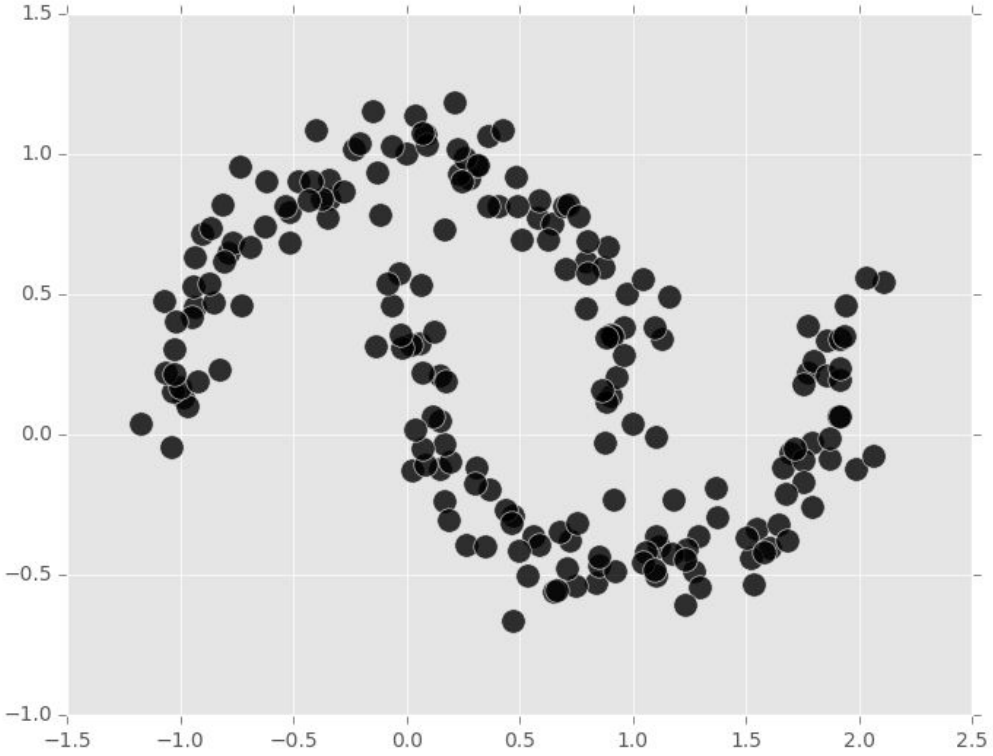
Density-Based Clustering in Python

Brian Kent
Dato, Inc

Dato
Create Intelligence™

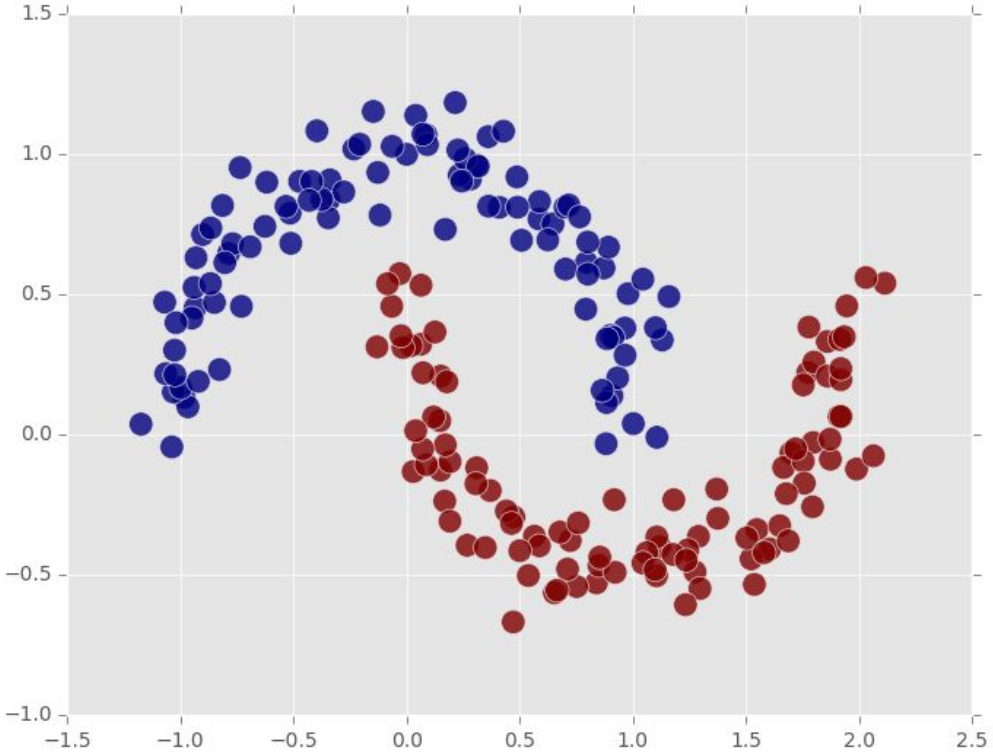
What is clustering?

- *Grouping data* instances
- *Similar* instances together
- No target variable



What is clustering?

- *Grouping data* instances
- *Similar* instances together
- No target variable



Why cluster?

- Wikipedia:
  - *"It is a main task of exploratory data mining, and a common technique for statistical data analysis"*

- ok, but why?
  - explore and visualize complex data
  - reduce data scale
  - detect outliers (and other anomalies)
  - deduplicate records
  - segment a market

Today's takeaways

- ***K-means*** isn't always a good option

- *Density-based clustering* is an alternative
  - ***DBSCAN*** is the most popular form
  - ***Level Set Trees*** are even more powerful

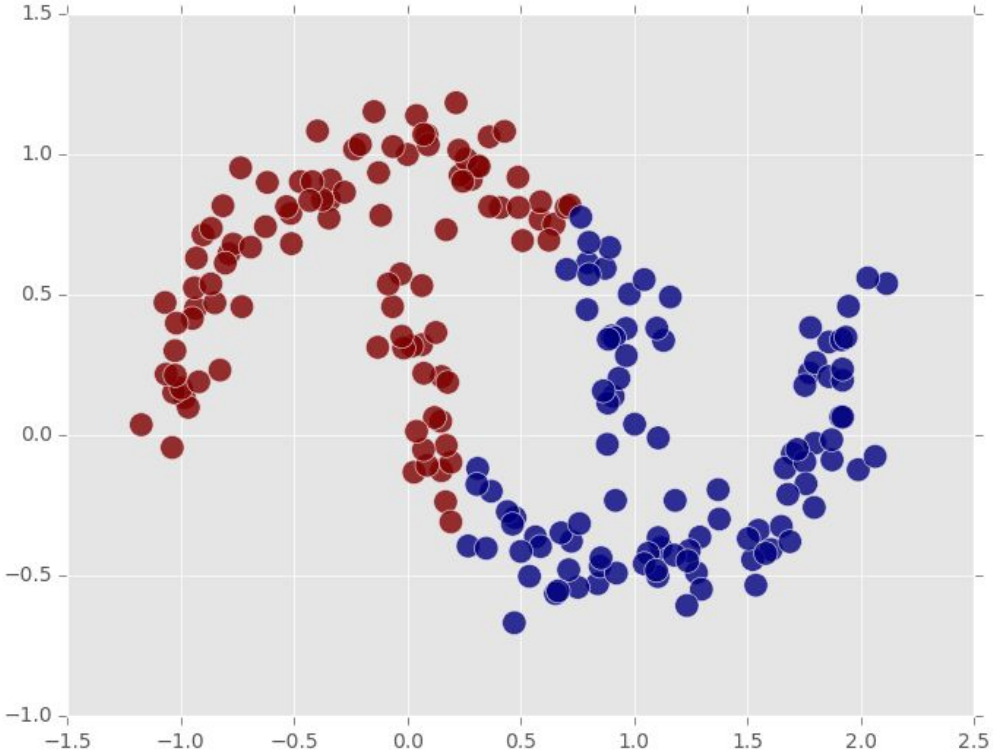
- Demos with **scikit-learn**, **GraphLab Create**, and **DeBaCL**

K-means is the default

- A very simple algorithm
- Lots of resources
- Lots of implementations
- Scales to very large datasets
  - Especially with the Elkan and minibatch implementations

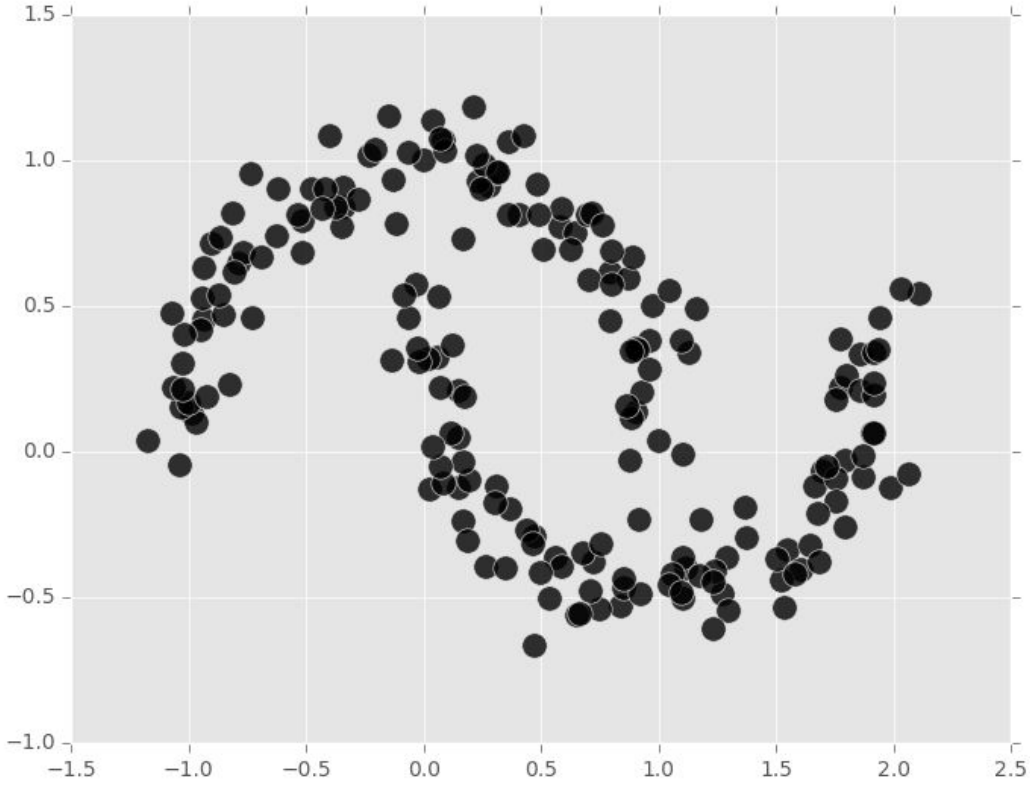
K-means isn't the only answer

- How to choose K?
- Sometimes choosing K is impossible.
- Spherical, convex clusters only.



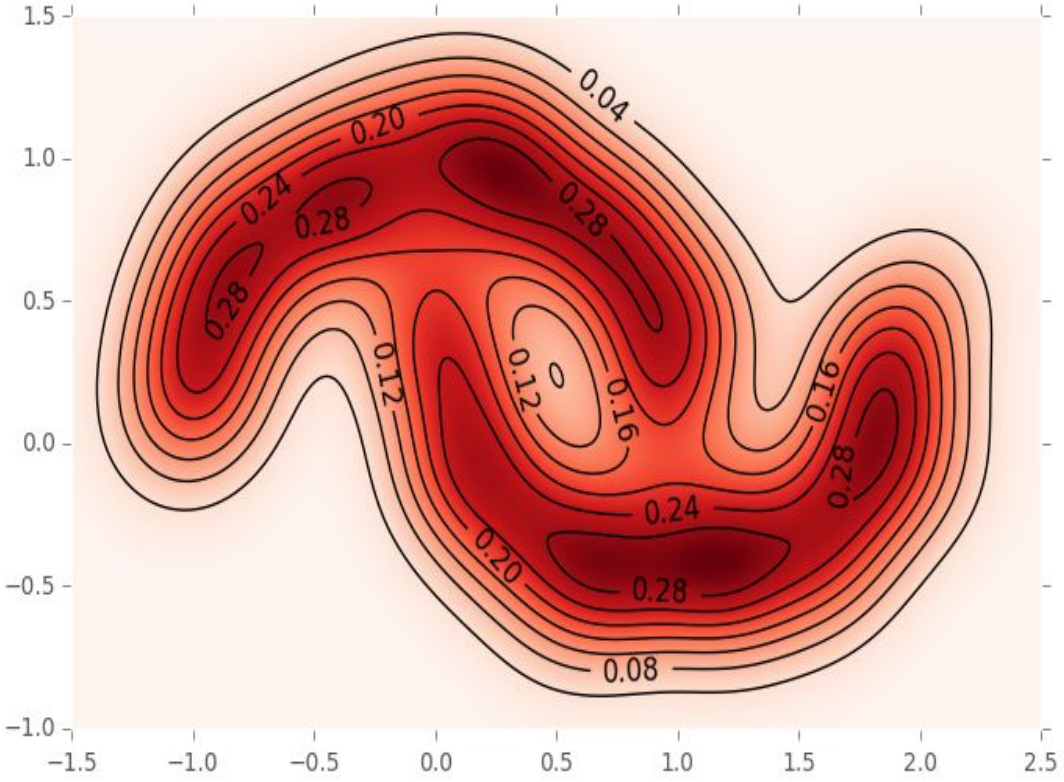
Enter density-based clustering

- ***Premise:*** data is drawn from a probability density function (PDF).



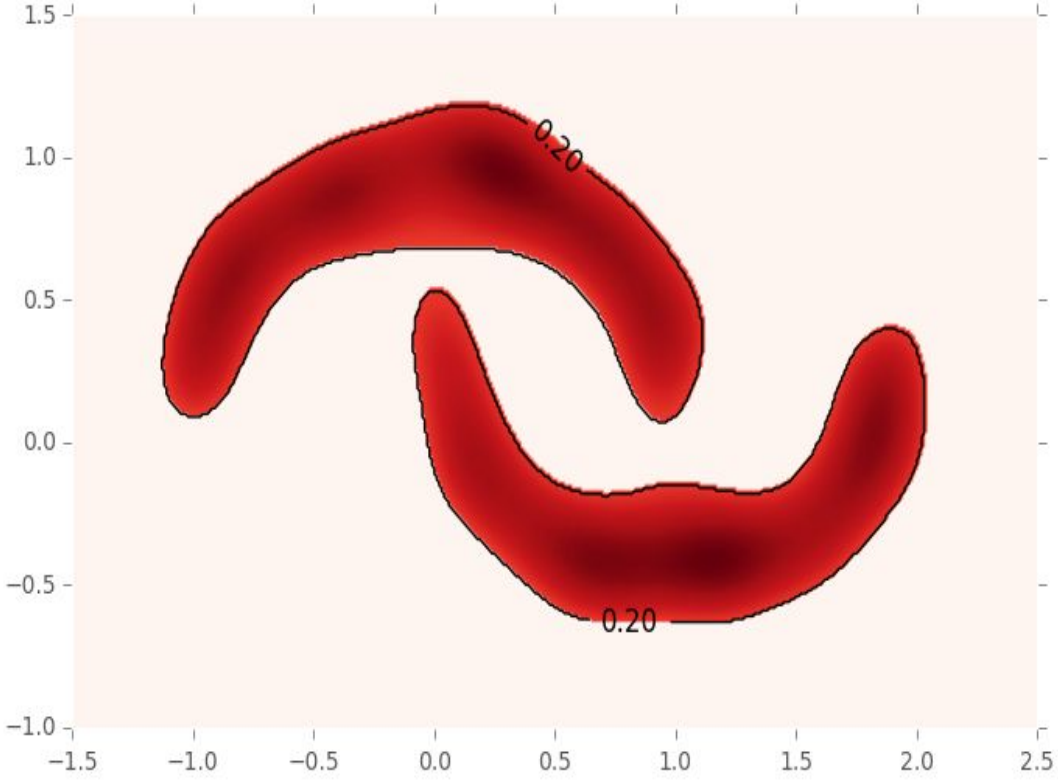
Enter density-based clustering

- ***Premise:*** data is drawn from a probability density function (PDF).
- Use the data to estimate the PDF.



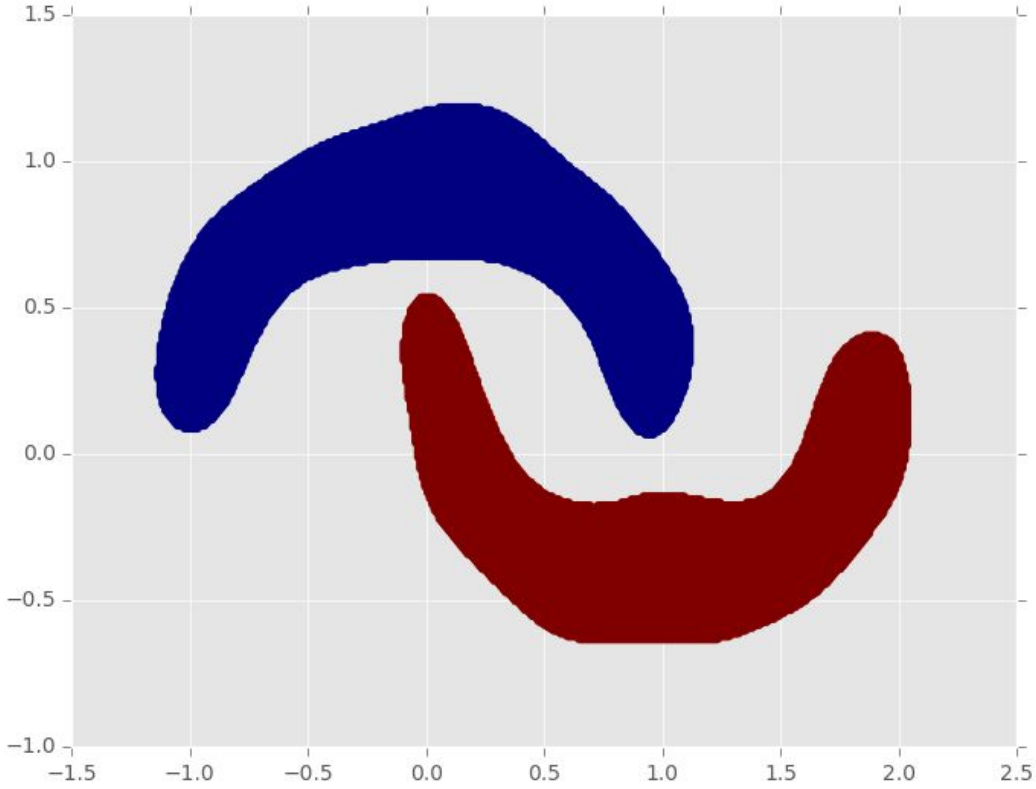
Enter density-based clustering

- Choose a threshold and get the *upper level set*.



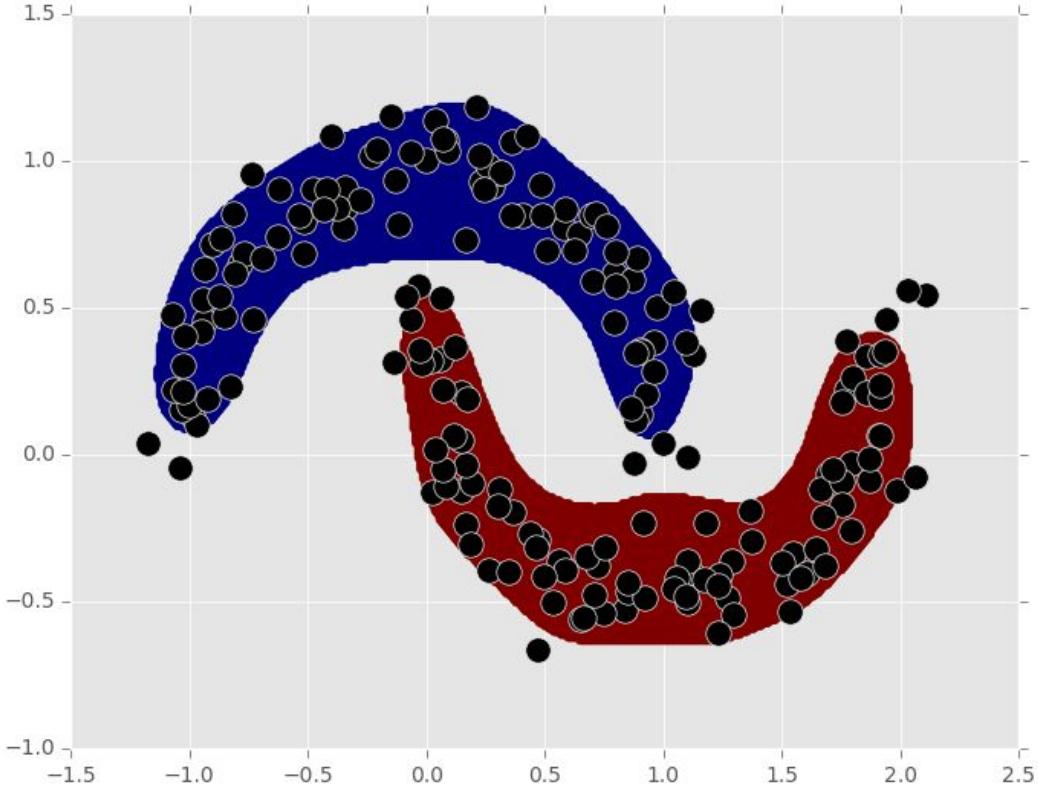
Enter density-based clustering

- Choose a threshold and get the *upper level set*.
- Find connected components of the upper level set.



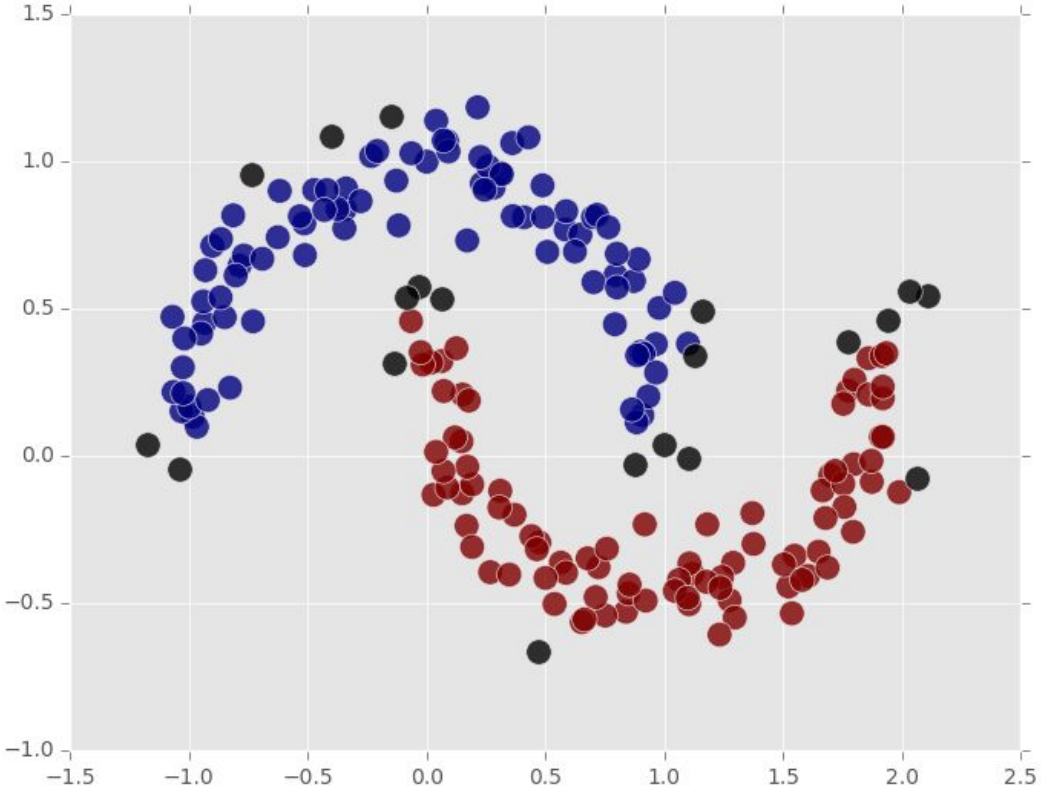
Enter density-based clustering

- Intersect the data with the connected components.



Enter density-based clustering

- Intersect the data with the connected components.
- Assign points to the corresponding cluster.



Pros and cons

- Recovers more complex cluster shapes.
- Don’t need to know K.
- Automatically find outliers.
- Requires a distance function.
- Not as scalable as K-means.
- …
- It’s impossible.
  - Can’t compute topologically connected components.

DBSCAN leads the pack

- ***Density-Based Spatial Clustering of Applications with Noise*** *(Ester, et al. 1996)*
- Test of Time award at KDD 2014.
- 7,400 citations on Google Scholar.
- **Main idea:**
  - three types of points: *core*, *boundary*, *noise*
  - connect core points into clusters
  - assign boundary points to clusters

DBSCAN Visualization

http://www.naftaliharris.com/blog/visualizing-dbscan-clustering/

Demo: DBSCAN in Scikit-learn

Notebook available at:

https://github.com/papayawarrior/public_talks

DBSCAN in GraphLab Create

- Built on scalable ***SFrame*** *and* ***SGraph*** data structures.
- ***Composite distances*** for varied feature types.
- Construct a similarity graph directly.
  - Permits a more efficient algorithm.
- Not open source, but free for non-commercial use.

Demo: DBSCAN in GraphLab Create

Notebook available at:

https://github.com/papayawarrior/public_talks

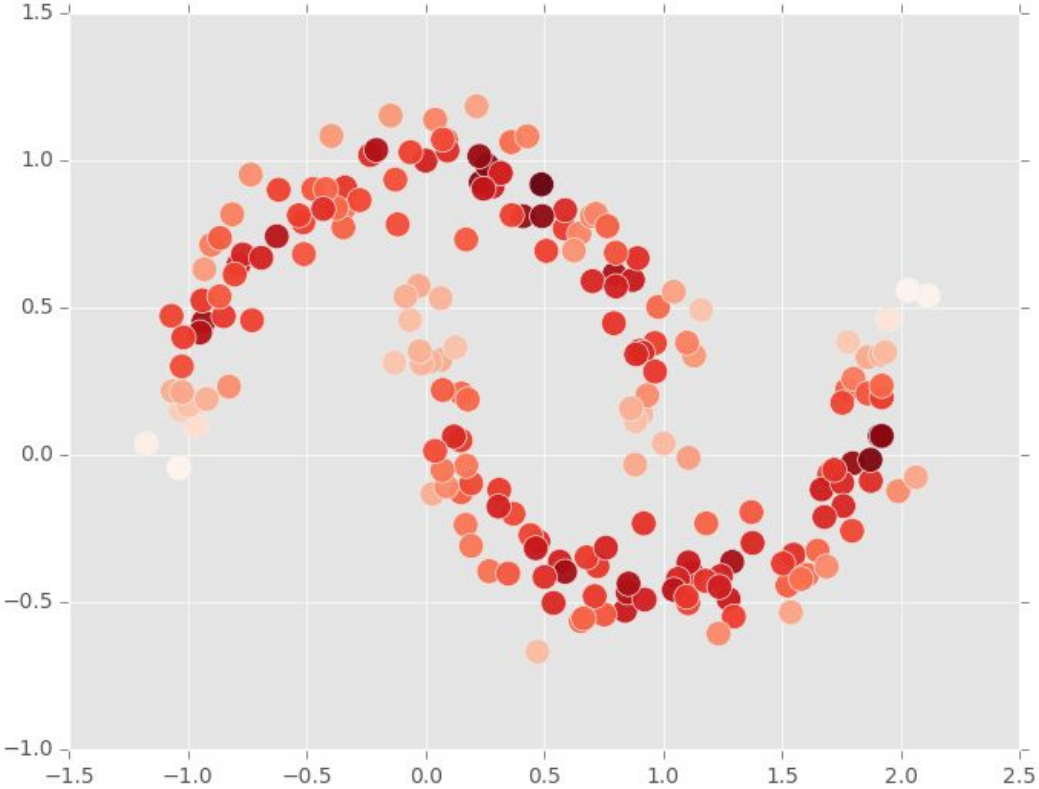
Level Set Trees are better than DBSCAN

- How to choose the density level (i.e.*min_neighbors*)?
- Changing levels means starting from scratch.
- ***Level Set Trees (LSTs)*** describe the entire hierarchy of density-based clusters.
  - Retrieve clusters in different ways without re-computing
  - Each cluster can have a *different* density level
  - Visualization of high-dim or complex data structure

Building a level set tree

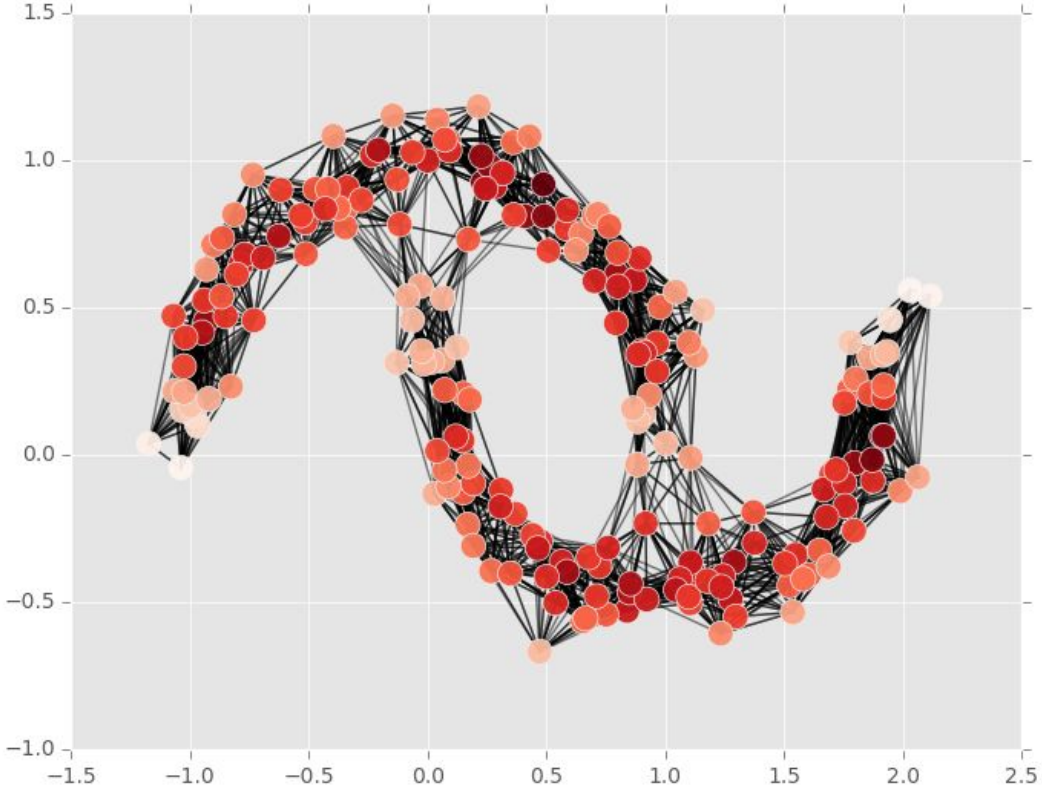
- Estimate the PDF at each data point.
- Construct a similarity graph on the data.
  - Vertices are data points.
  - Edges represent near neighbors.
- Remove vertices in order of estimated density.
- Compute the connected components at each level.
  - Keep track of components *between* levels.

Building a level set tree



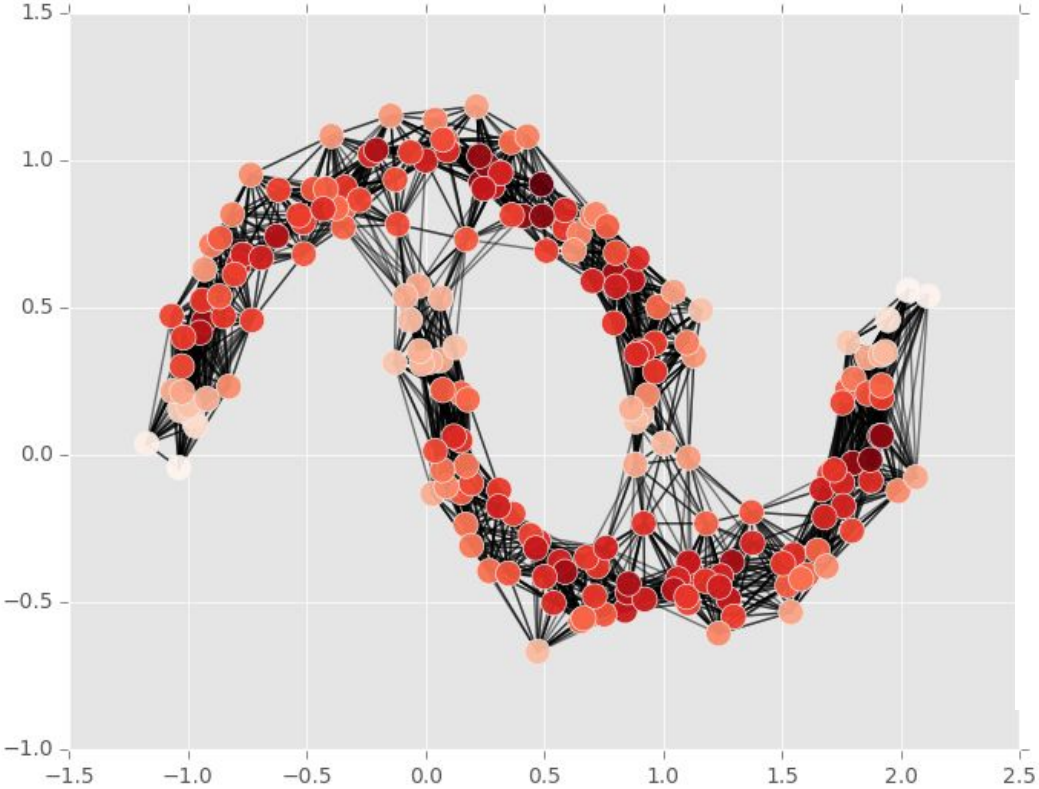
- Estimate the density.

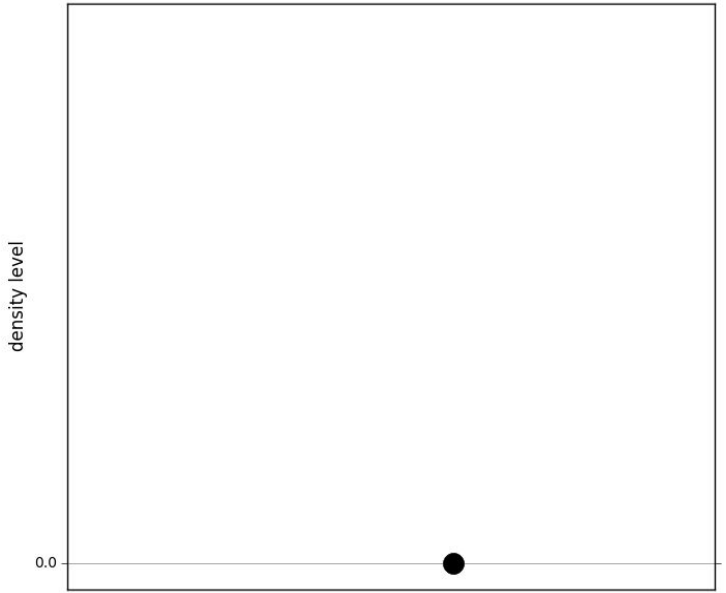
Building a level set tree



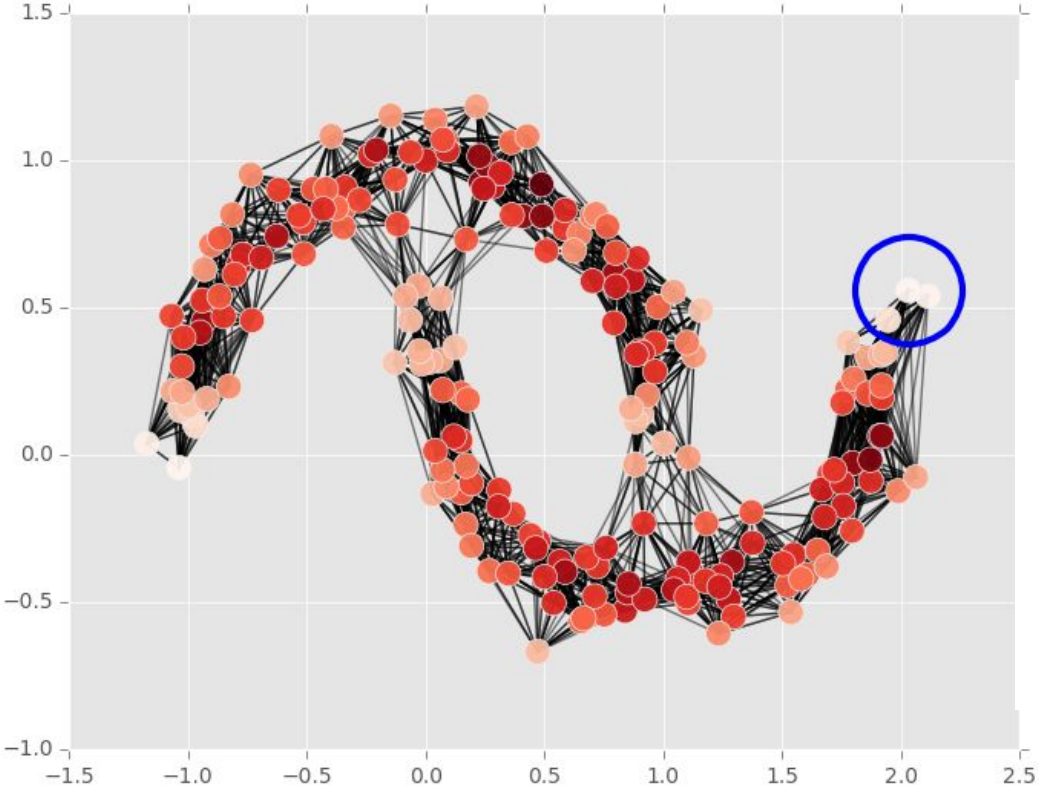
- Estimate the density.
- Construct a similarity graph.

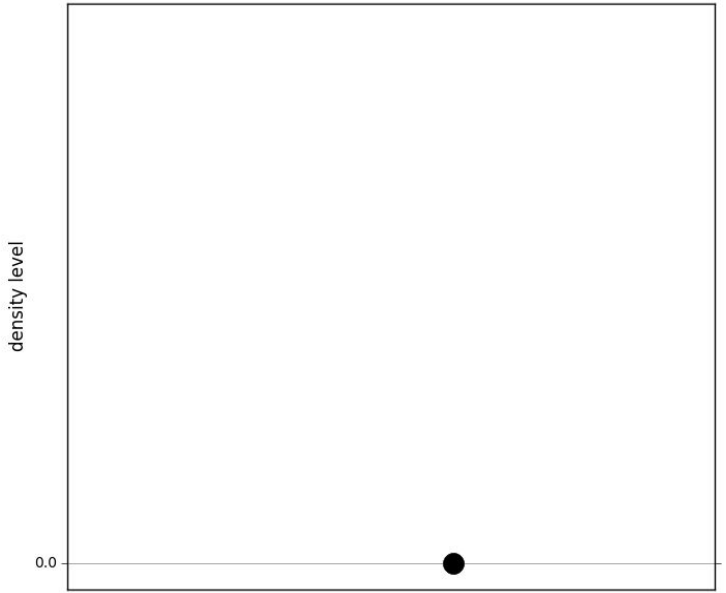
Building a level set tree



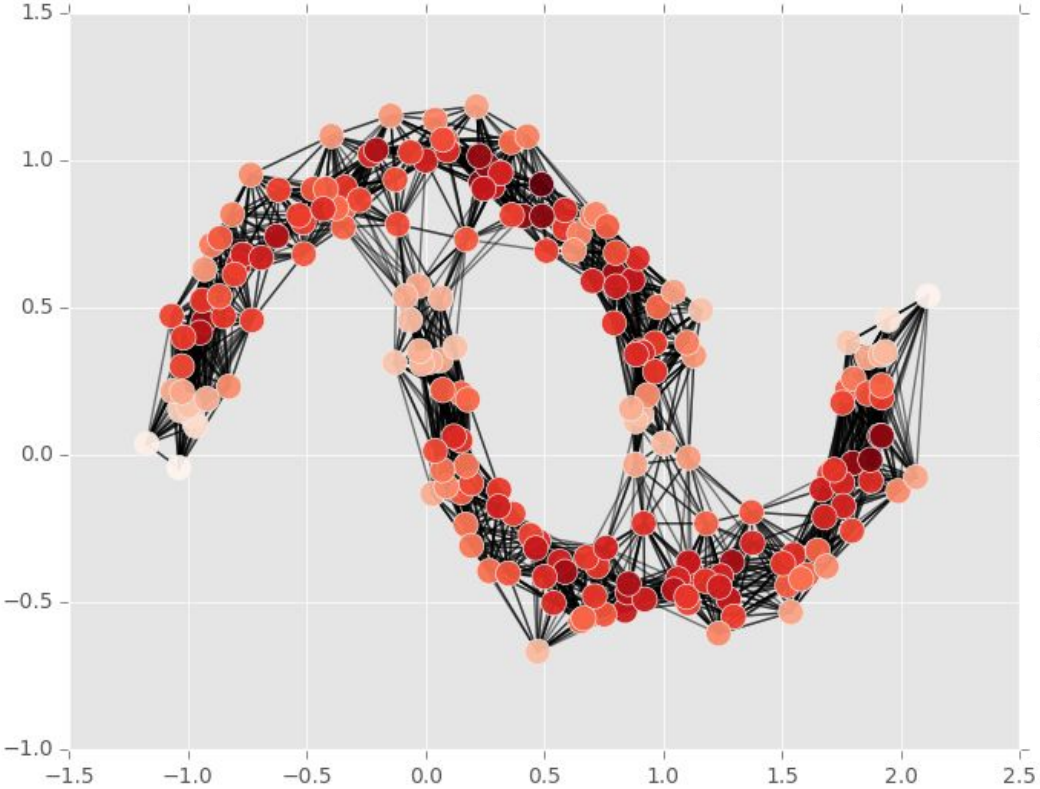


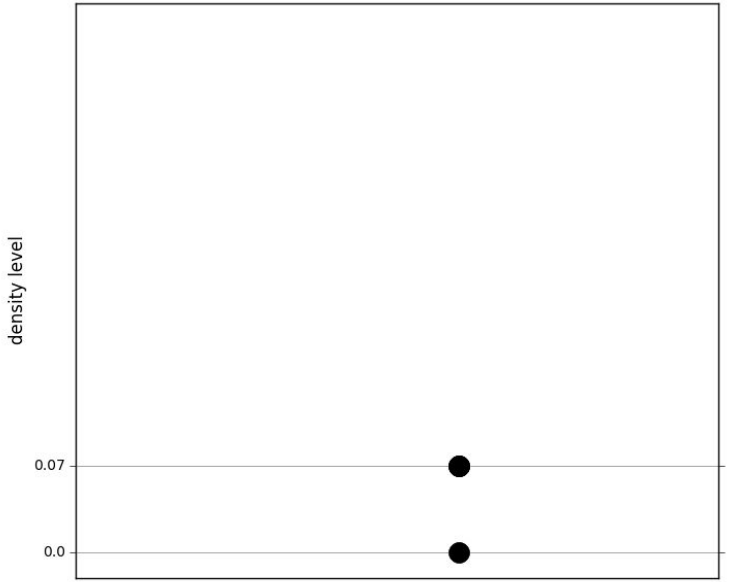
Building a level set tree



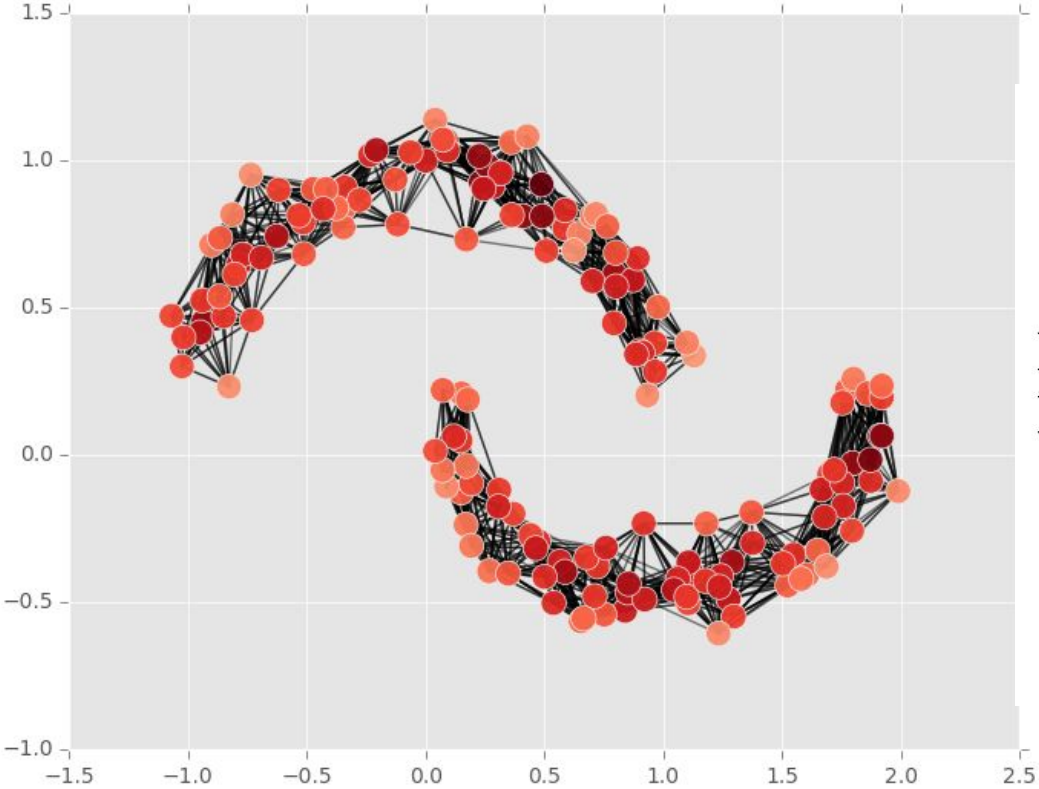


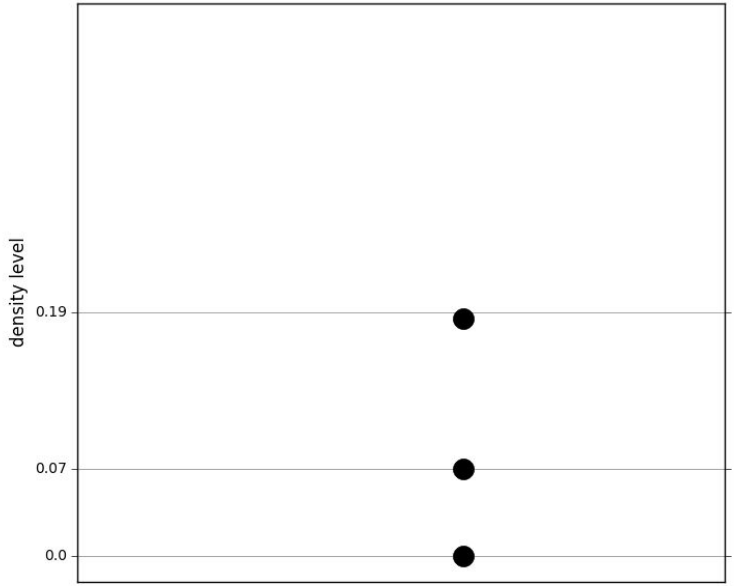
Building a level set tree



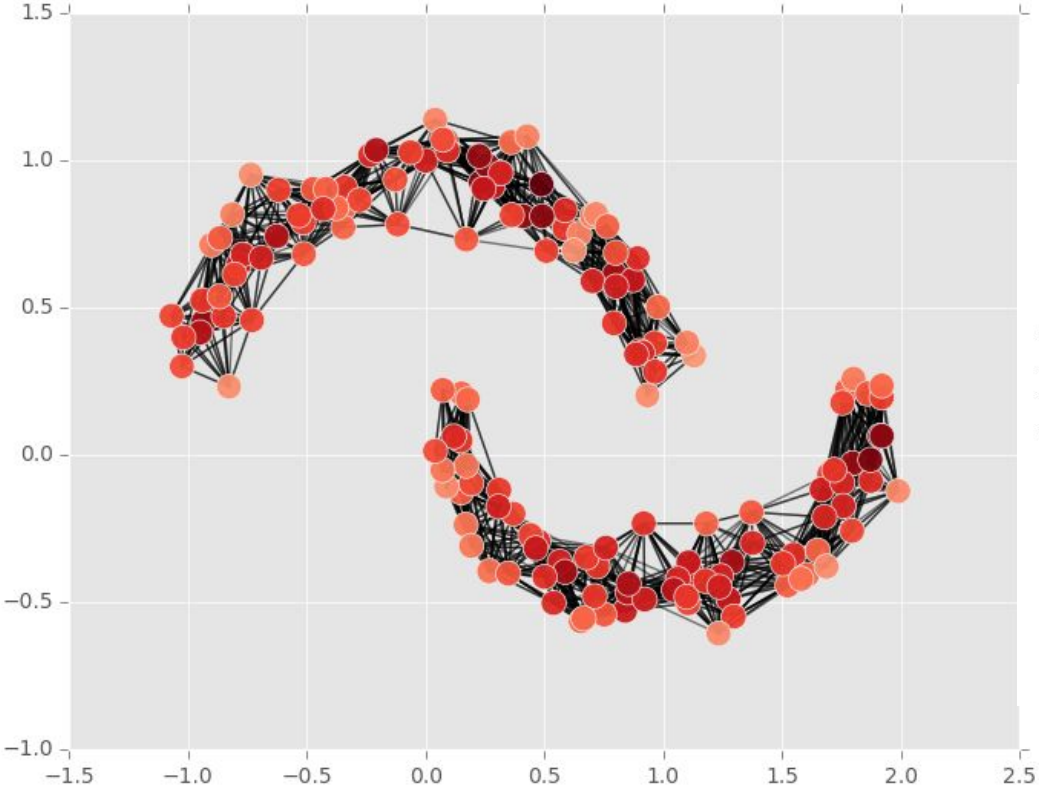


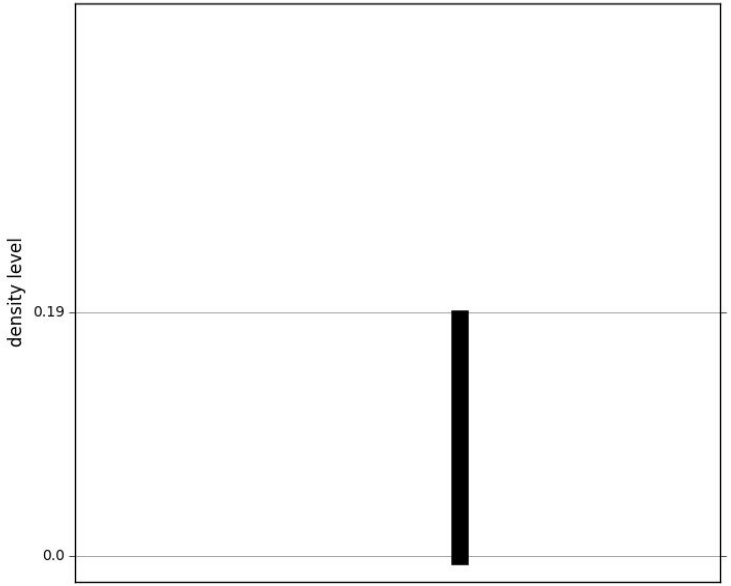
Building a level set tree



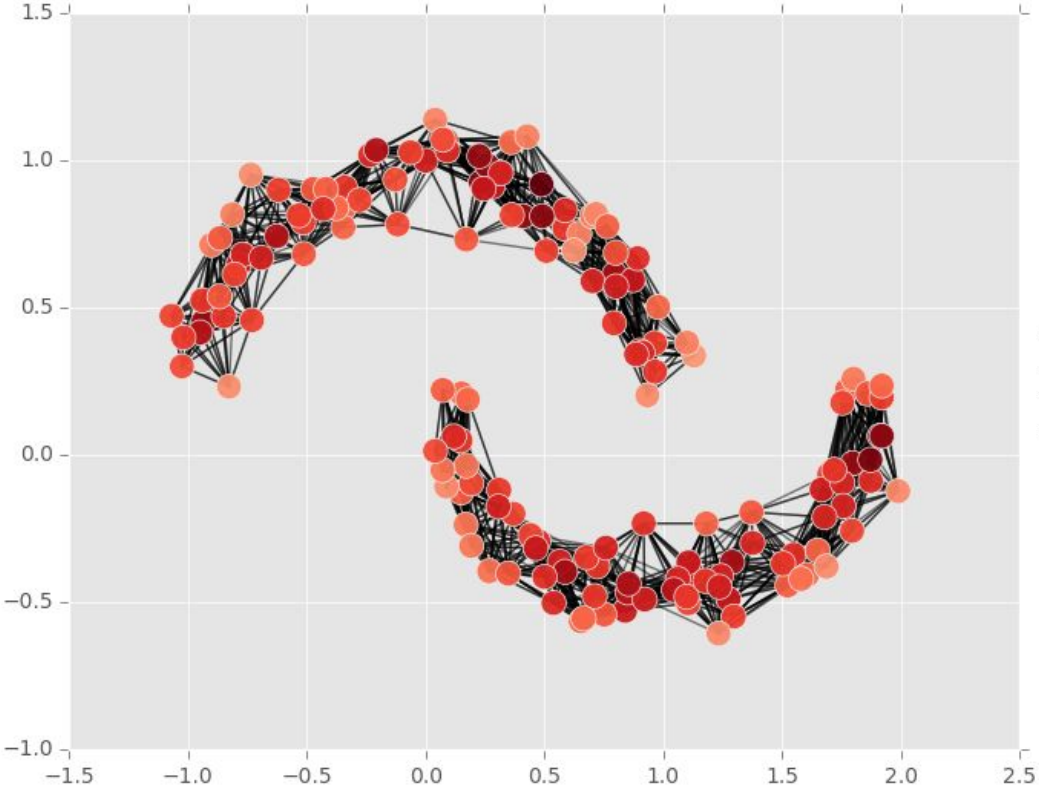


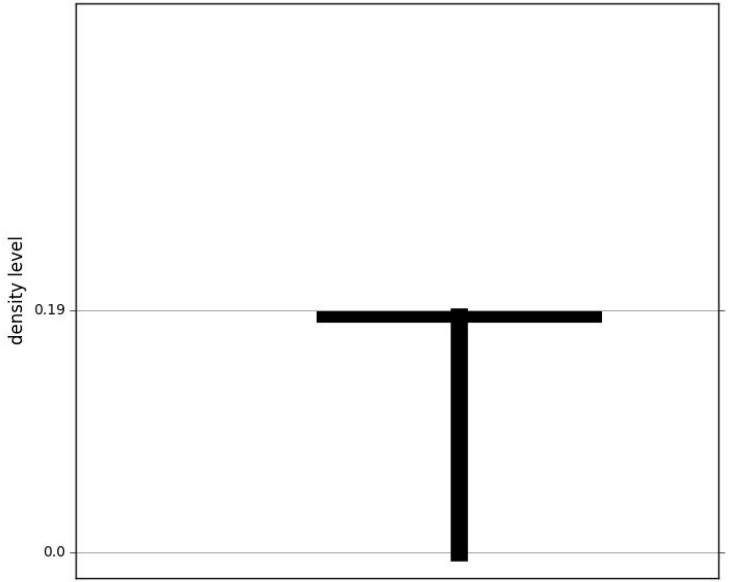
Building a level set tree



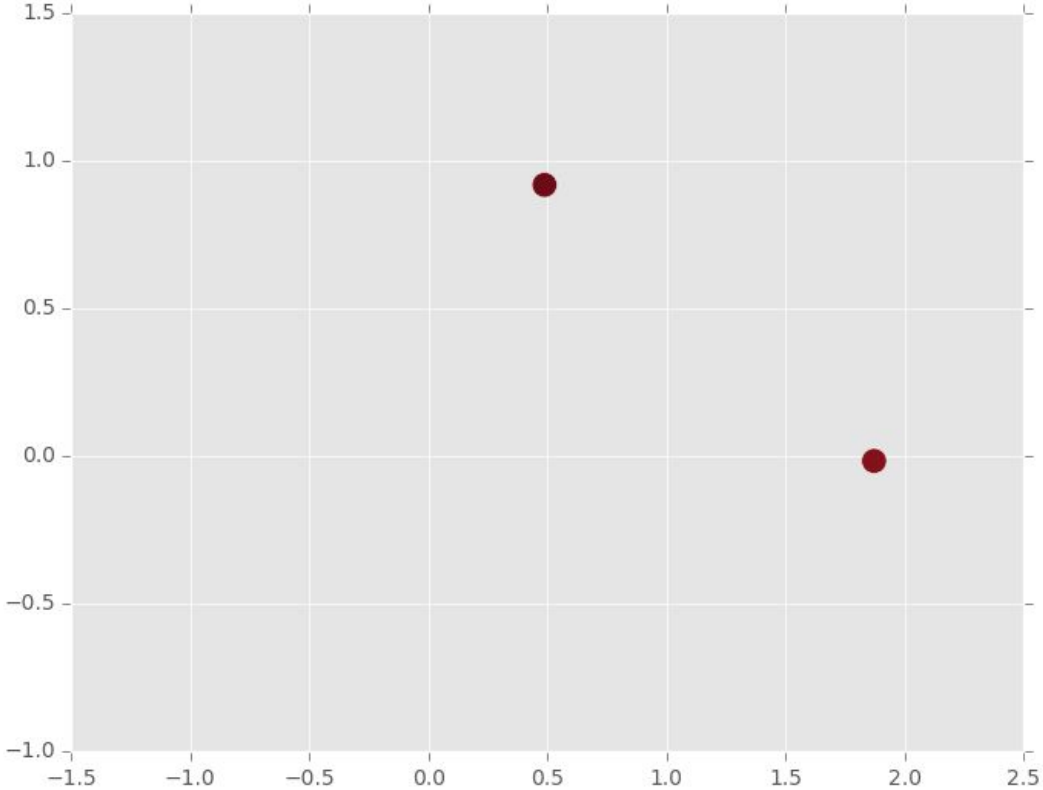


Building a level set tree

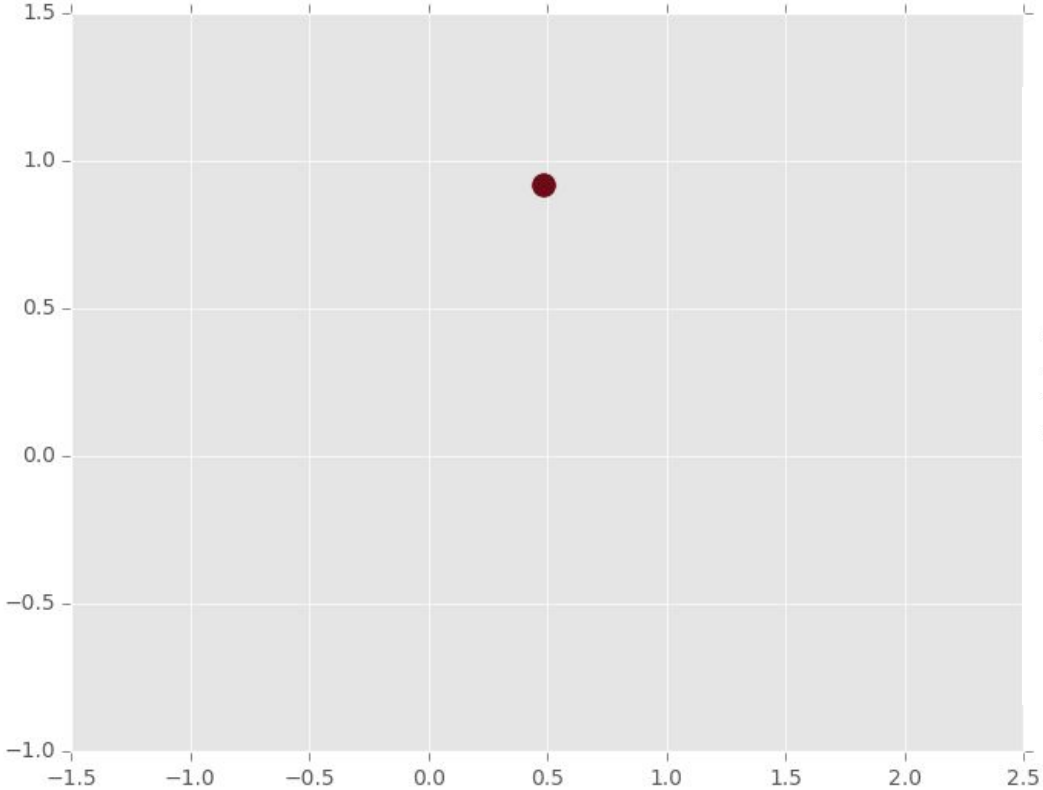


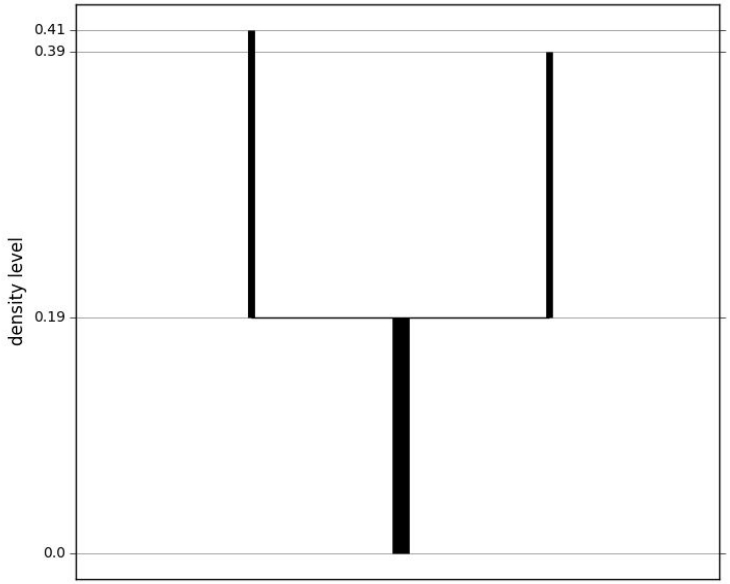


Building a level set tree



Building a level set tree



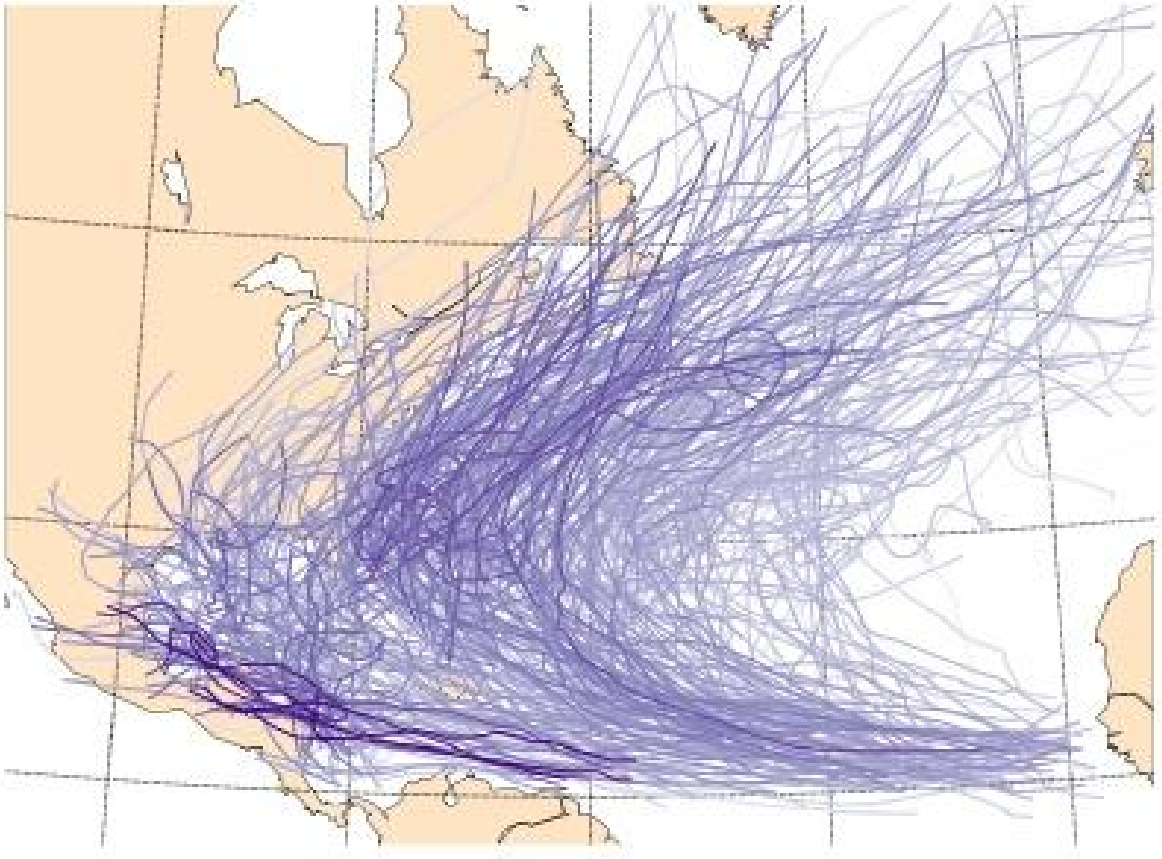


Demo: level set trees with DeBaCl

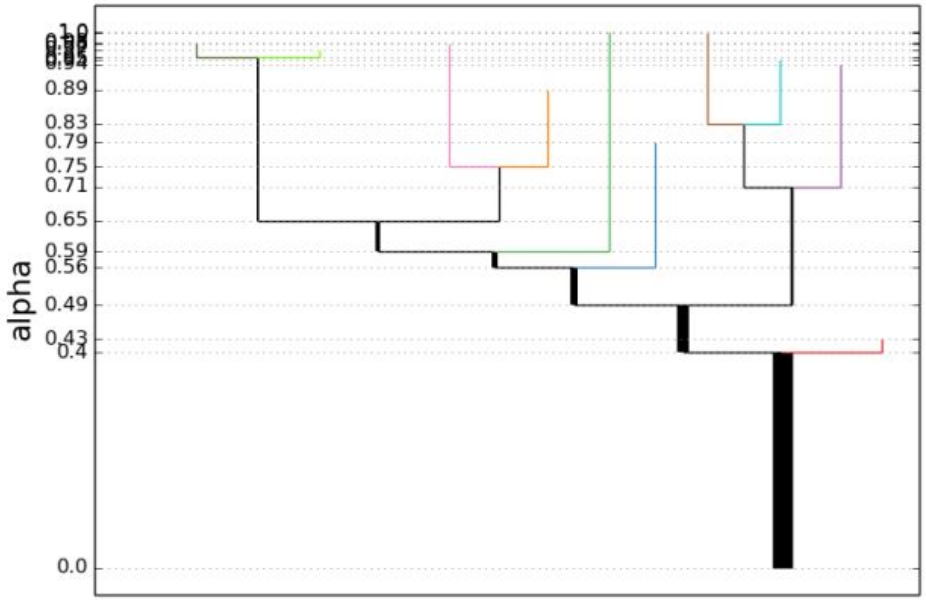
Notebook available at:

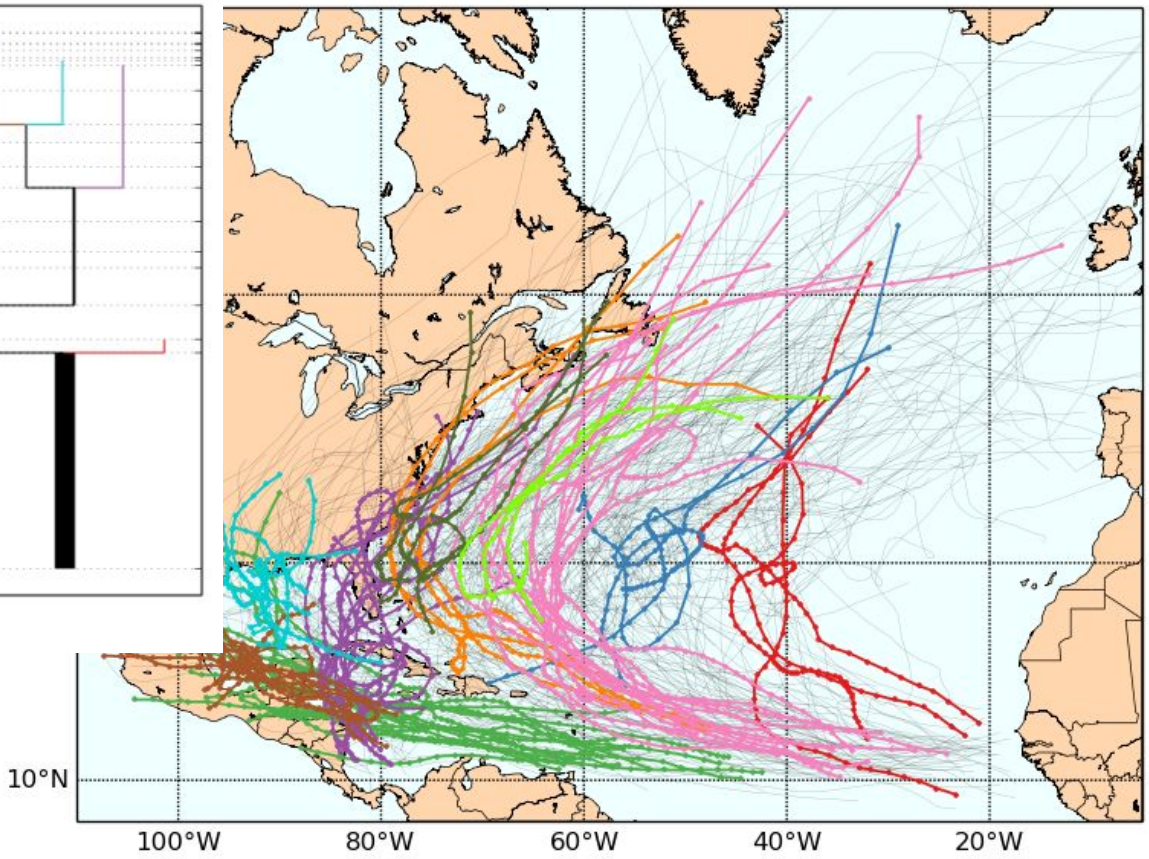
https://github.com/papayawarrior/public_talks

LSTs shine with complex data



LSTs shine with complex data





DeBaCl builds level set trees

- ***DeBaCl****:* DEnsity-BAsed CLustering
- *pip install debacl*
- https://github.com/coaxlab/debacl
- Help wanted!

Wrap-up

- K-means isn't always the best option.
- Density-based clustering can be a good alternative.
- DBSCAN is the most popular form.
  - ***Scikit-learn, GraphLab Create***
- Level set trees are even more powerful.
  - ***DeBaCL***
  - Help on DeBaCl is very welcome!

Thanks!