BI Over Petabytes: Meet Apache Mahout

Industrial Strength Machine Learning

April 2009

http://lucene.apache.org/mahout/

BI and ML

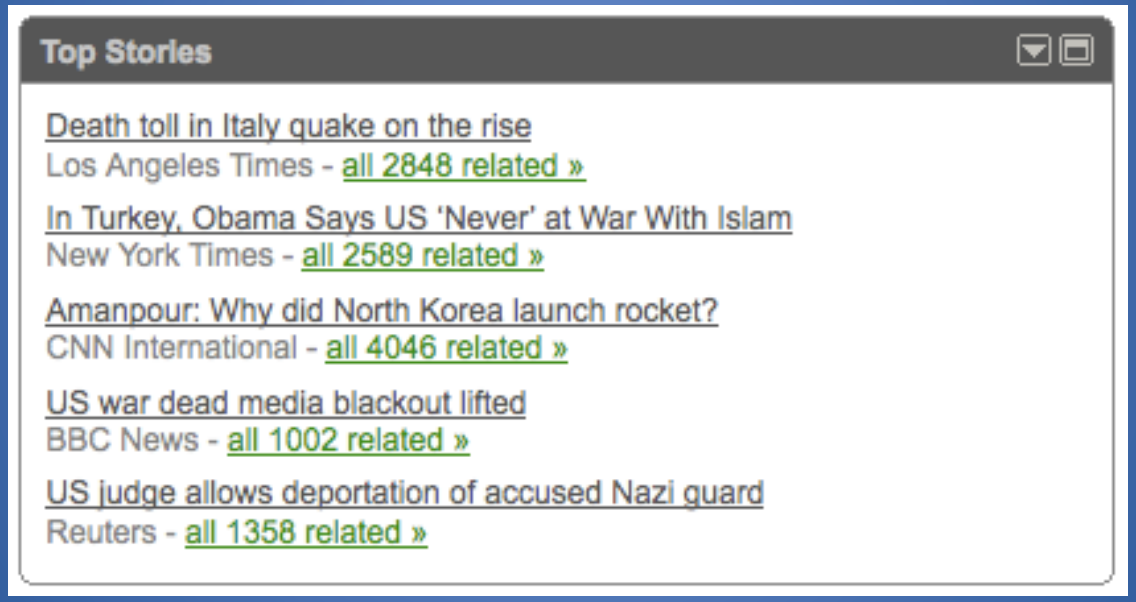
- Business Intelligence
  - OLAP
  - Analytics
  - Data mining
  - Performance analysis
  - Text mining
  - Predictive analysis
- Machine Learning
  - Classification
  - Clustering
  - Regression
  - Collaborative filtering
  - Evolutionary algorithms

What is Machine Learning?

- "Machine learning is the subfield of artificial intelligence that is concerned with the design and development of algorithms that allow computers to improve their performance over time …"
  (http://en.wikipedia.org/wiki/Machine_learning)
- Types of ML algorithms
  - Supervised: Using labeled training data, create a function that predicts output for unseen inputs
  - Unsupervised: Using unlabeled data create a function that can predict output
  - Semi-supervised: Uses labeled and unlabeled data

One Common ML Example

Text Clustering



Google.com

Another Common Example

Collaborative Filtering

Customers Who Bought This Item Also Bought

Pattern Recognition and Machine Learning (Information Sci... by Christopher M. Bishop

★★★★☆ (41) $58.86

The Elements of Statistical Learning by T. Hastie

★★★★☆ (27) $75.17

Programming Collective Intelligence: Building Smart Web 2... by Toby Segaran

★★★★½ (34) $26.39

Introduction to Data Mining by Pang-Ning Tan

★★★★★ (10) $87.97

Amazon.com

Where ML is Used Today

- Internet search clustering
- Knowledge management systems
- Social network mapping
- Taxonomy transformations
- Marketing analytics
- Recommendation systems
- Log analysis & event filtering
- SPAM filtering, fraud detection

Current Situation

- Vast amounts of data are now available via the Internet
- Platforms now exist to run computations over large datasets (MapReduce, Hadoop, Dryad)
- Sophisticated analytics are needed to turn data into information people can use
- Active Machine Learning research community and research/proprietary implementations of ML algorithms
- The world needs scalable implementations of ML under open license - ASF

History of Mahout

- Summer 2007
  - Developers needed scalable ML
  - Mailing list formed
- Community formed
  - Apache contributors
  - Academia & industry
  - Lots of initial interest
- Mahout project formed under Apache Lucene
  - January 25, 2008
  - Mahout 0.1 release April, 2009

Who We Are (so far)

Grant Ingersoll

Dawid Weiss

Ozgur Yilmazel

Erik Hatcher

Karl Wettin

Jeff Eastman

Ted Dunning

Sean Owen

Otis Gospodnetic

Isabel Drost

Release 0.1 Code Base

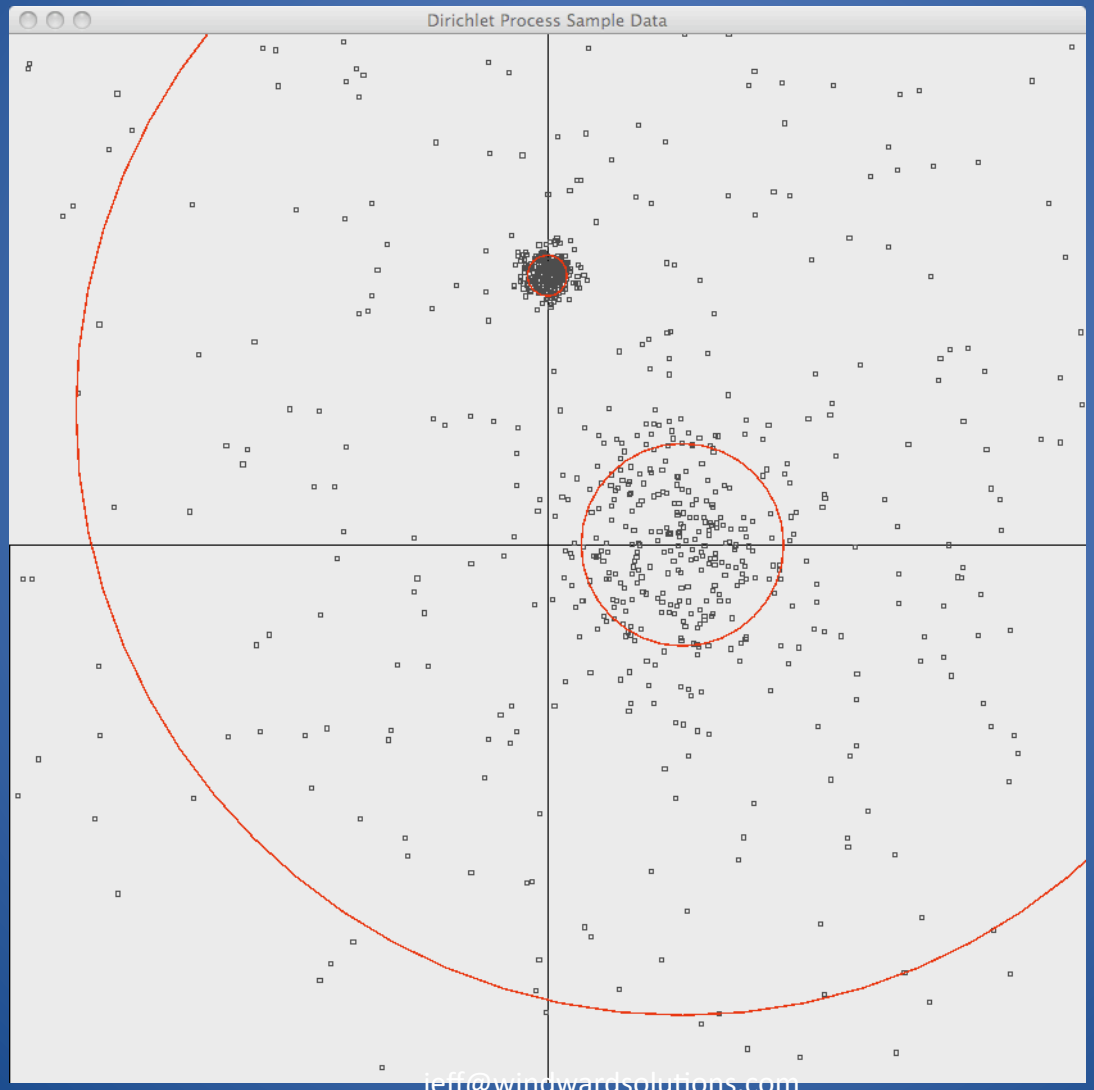
- Matrix & Vector library
  - Memory resident sparse & dense implementations
- Classification
  - Naïve Bayes, Complementary Naïve Bayes
- Clustering
  - Canopy
  - K-Means, fuzzy K-Means
  - Mean Shift
  - Dirichlet Process
- Collaborative Filtering
  - Taste
- Evolutionary Algorithms
  - Watchmaker
- Utilities
  - Distance Measures
  - Parameters

Highly scalable, parallel implementations on the Apache Hadoop platform

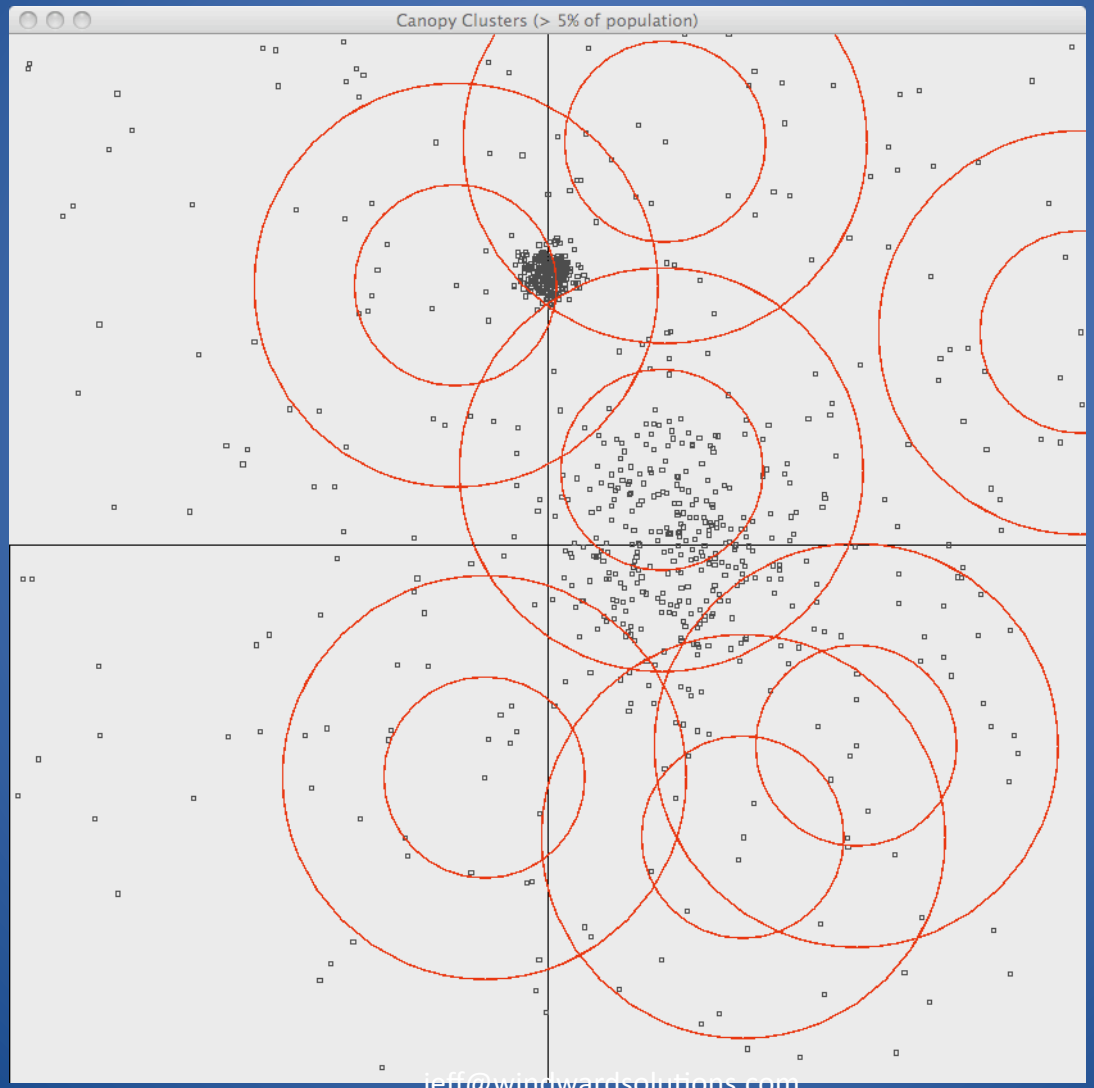
Examples: Clustering

- Canopy
  - Single pass (fast approximation) assigns every point to a single cluster
  - *Inputs: Distance Measure*, T1, T2 canopy values
- Mean Shift
  - Iterative process converges on modes of density distribution
  - Inputs: Distance Measure, T1, T2 values, convergence criteria
- K-Means
  - Iterative process converges on a single, 'best' assignment of points to clusters
  - *Inputs: Distance Measure*, initial clusters, convergence criteria
- Fuzzy K-Means
  - Like K-Means but uses probability density function to weight all points against all clusters
- Dirichlet Process
  - Bayesian: incorporates prior domain knowledge as a mixture of models
  - Iterative process converges on multiple, 'most likely' answers
  - Inputs:
    - Number of models, number of iterations to perform
    - *Model* (parameters, observations, probability density function)
    - *Model Distribution* (prior, posterior sampling)

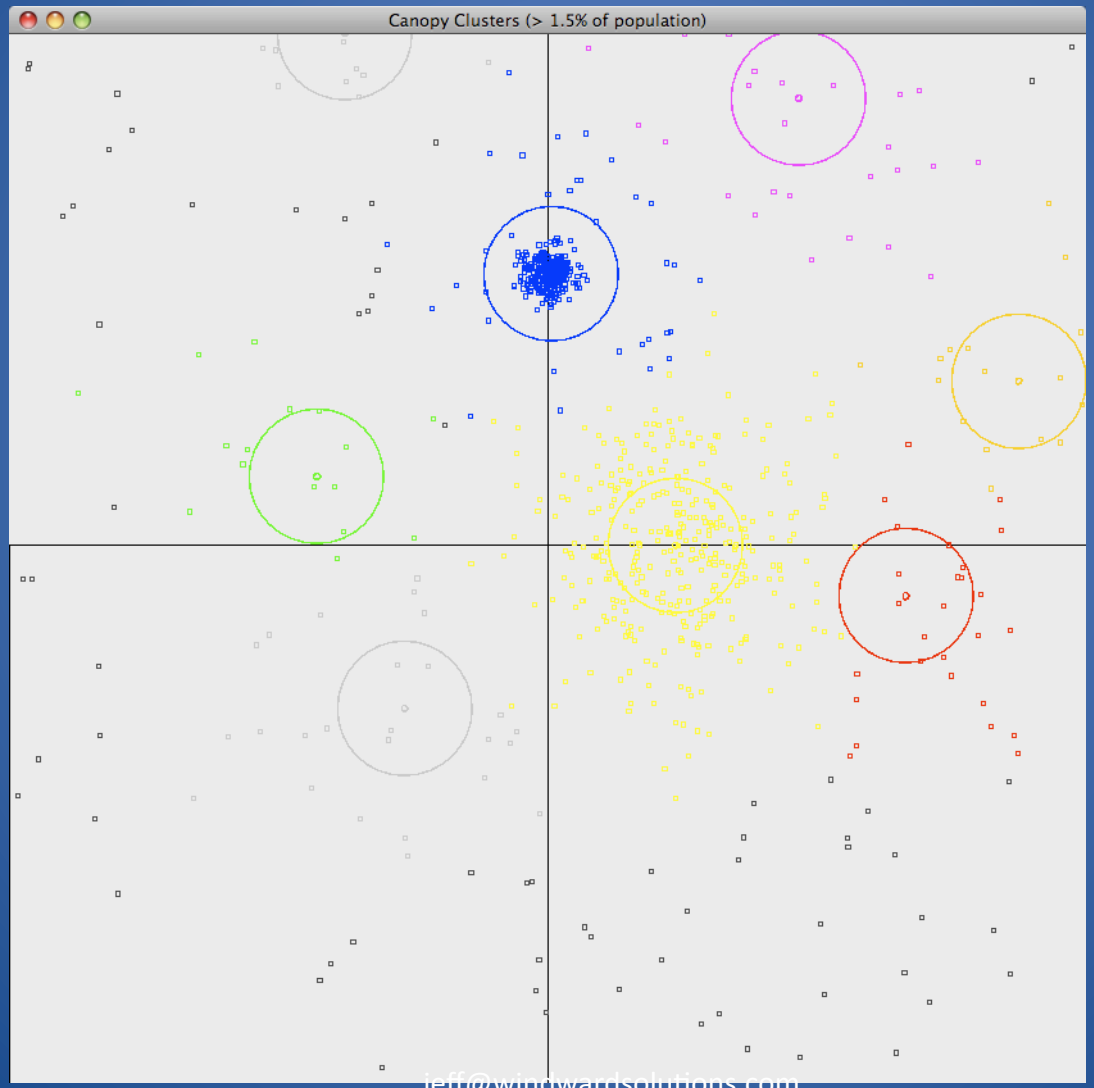
Sample Data



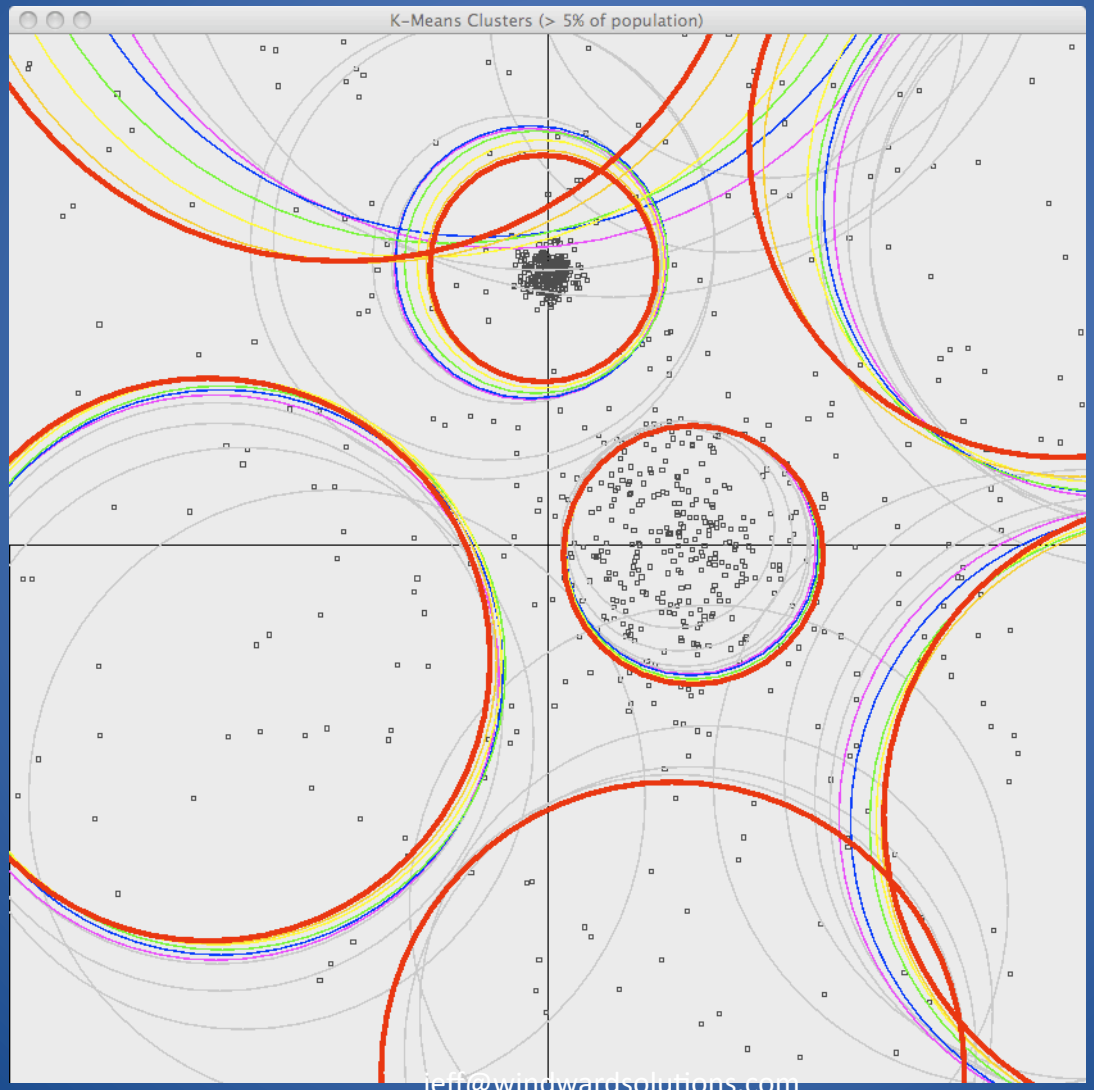
Canopy Clusters



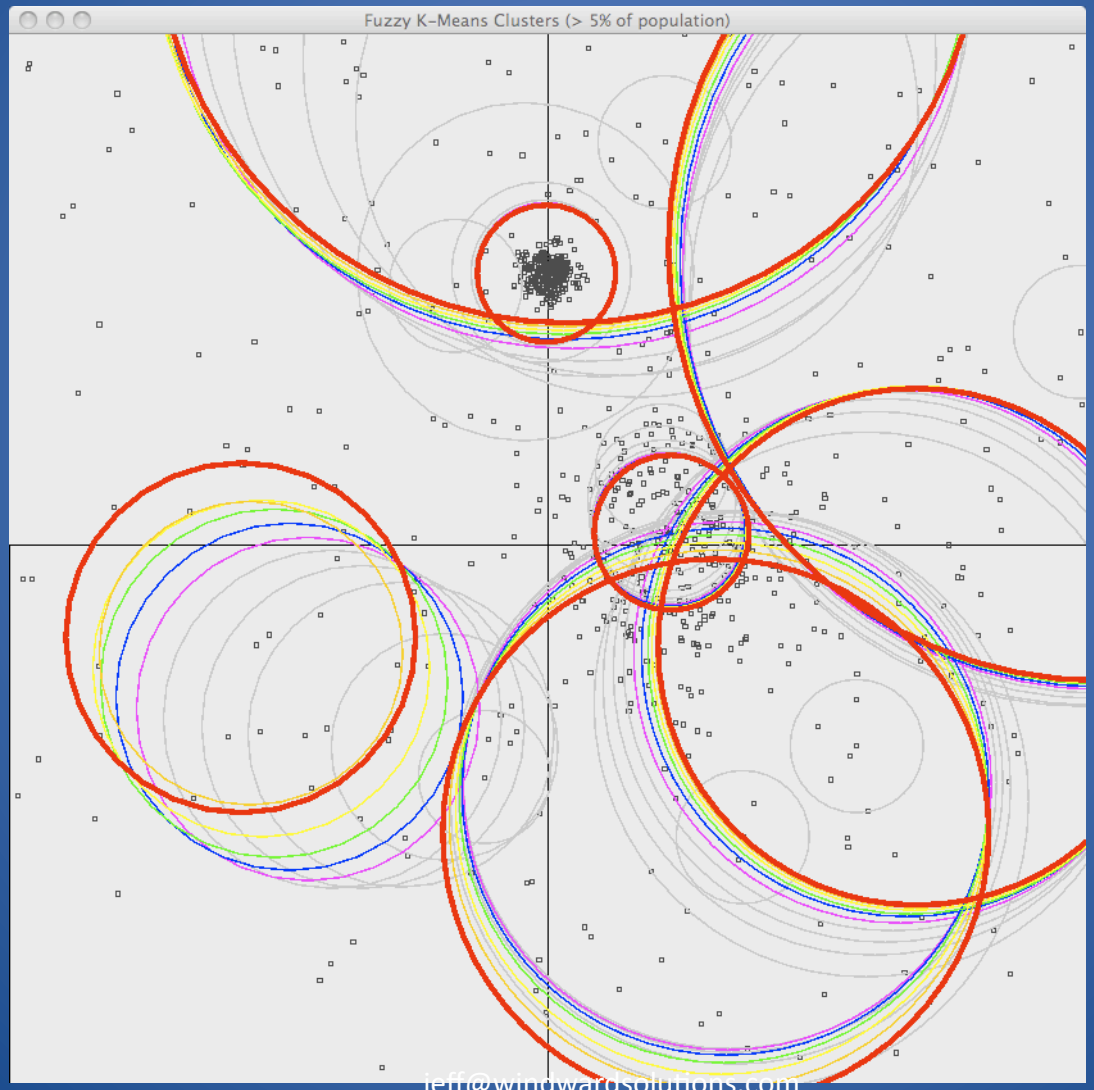
Mean Shift Clusters



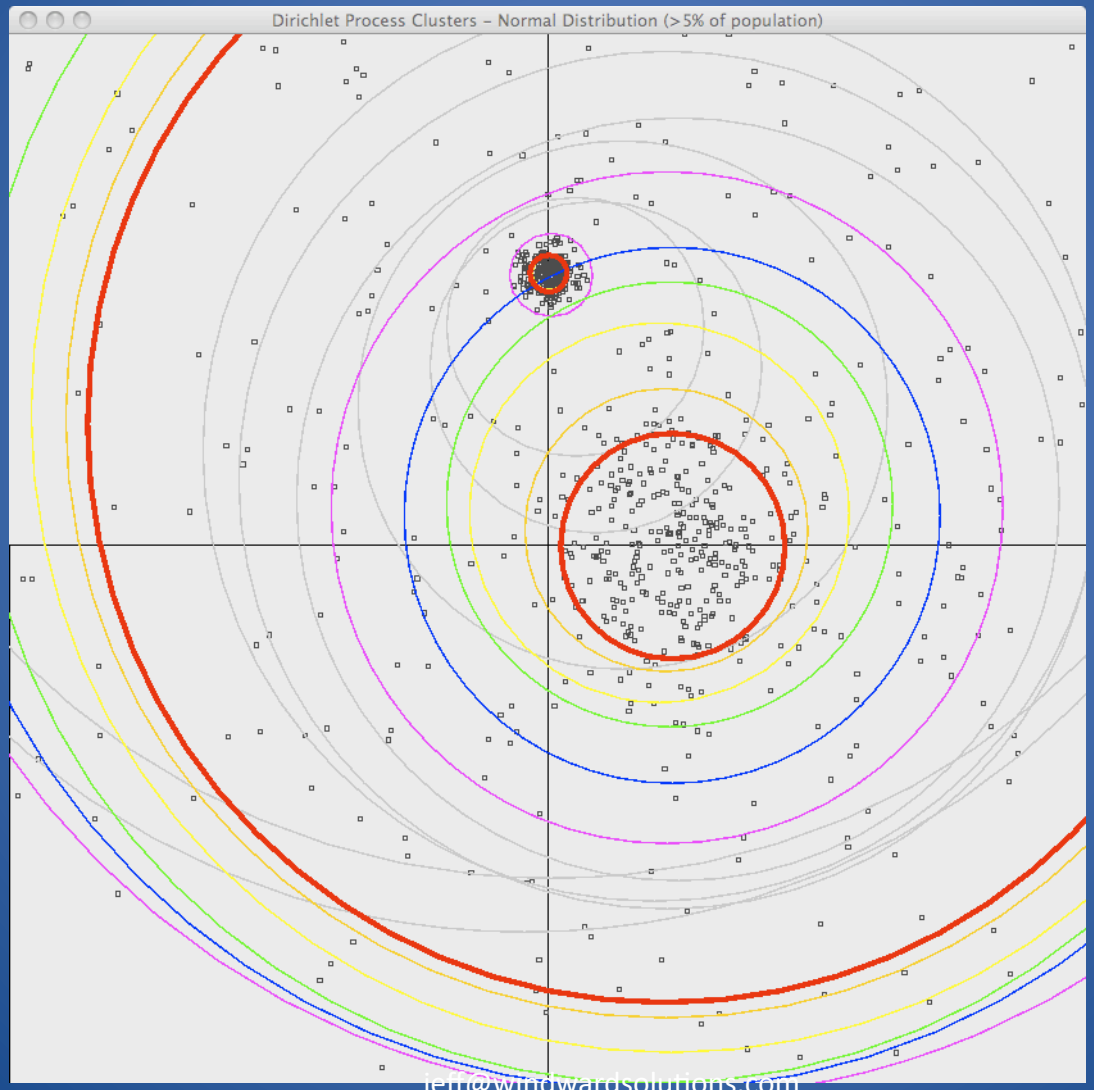
K-Means Clusters



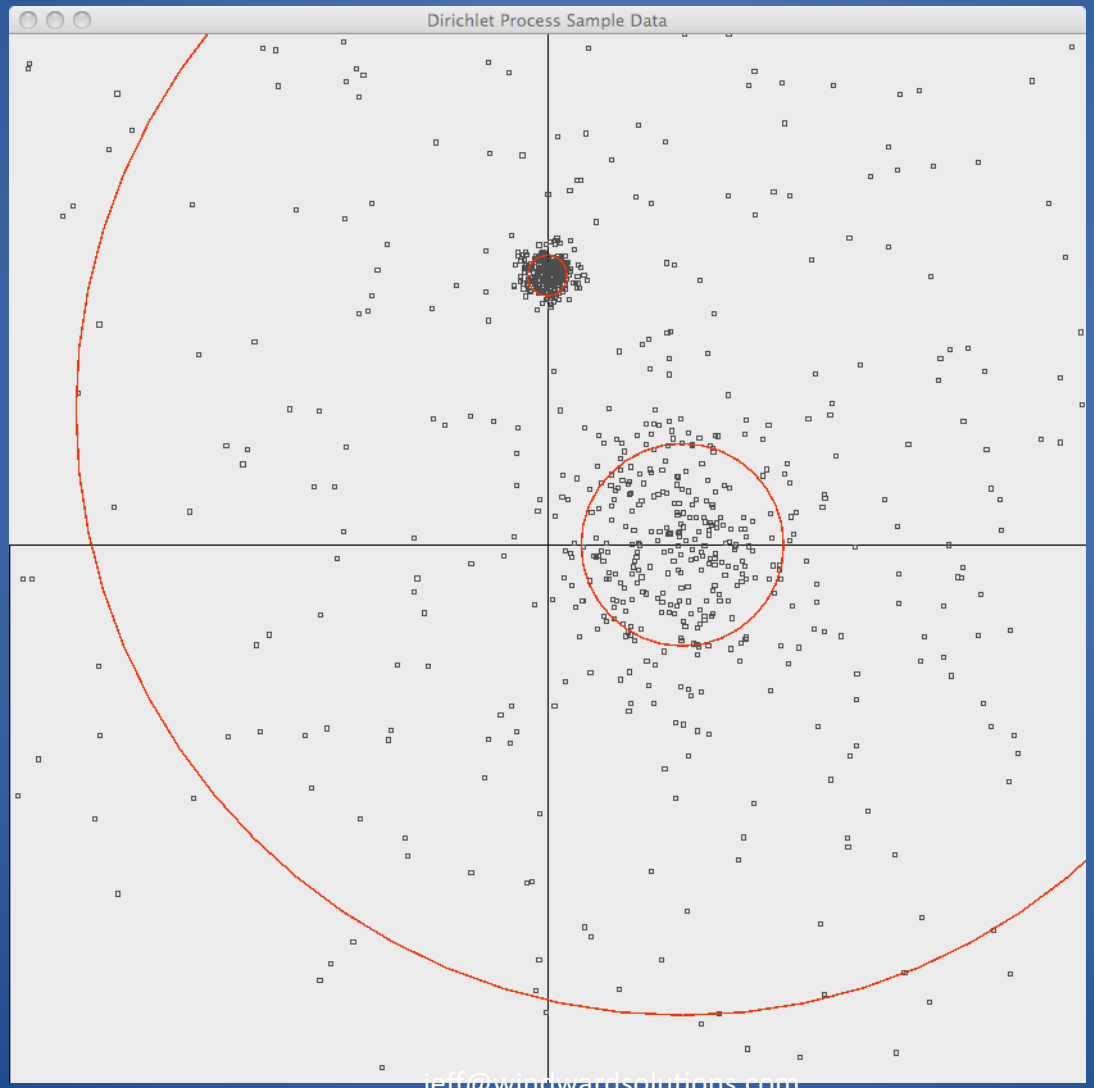
Fuzzy K-Means Clusters



Dirichlet Process Clusters



Sample Data (Again)

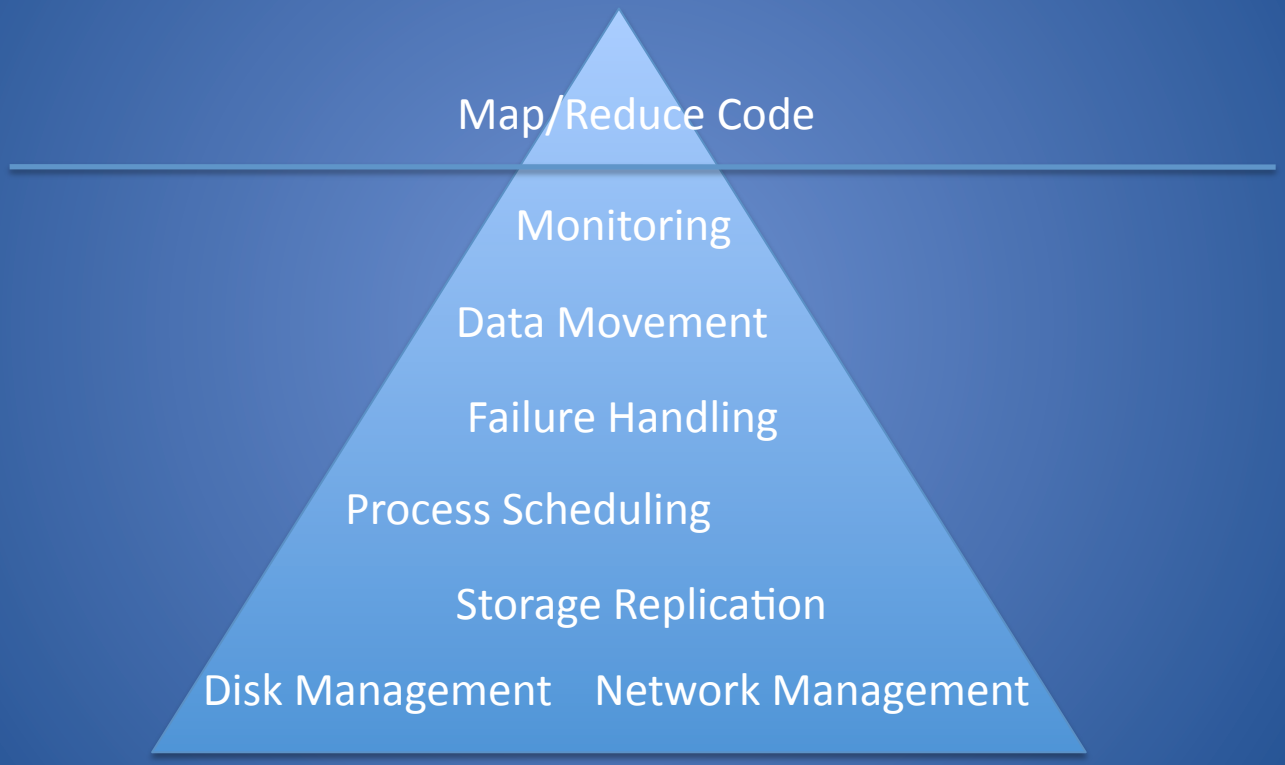


Apache Hadoop

http://hadoop.apache.org

- Uses clusters of (5-10,000) general purpose Linux boxes
- HDFS supports redundant file storage and streaming access in the face of predictable hardware failures
- Map/Reduce API simplifies programming of algorithms that operate over vast datasets
- Hbase offers Google BigTable style of schema-less, temporal database
- PIG offers higher level language for manipulating very large datasets that reduces the need for M/R programming
- Zookeeper is a highly available and reliable coordination system used to synchronize state between applications
- Hive is a data warehouse infrastructure that provides data summarization, adhoc querying and analysis of datasets

The Hadoop Iceberg



(http://hadoop.apache.org)

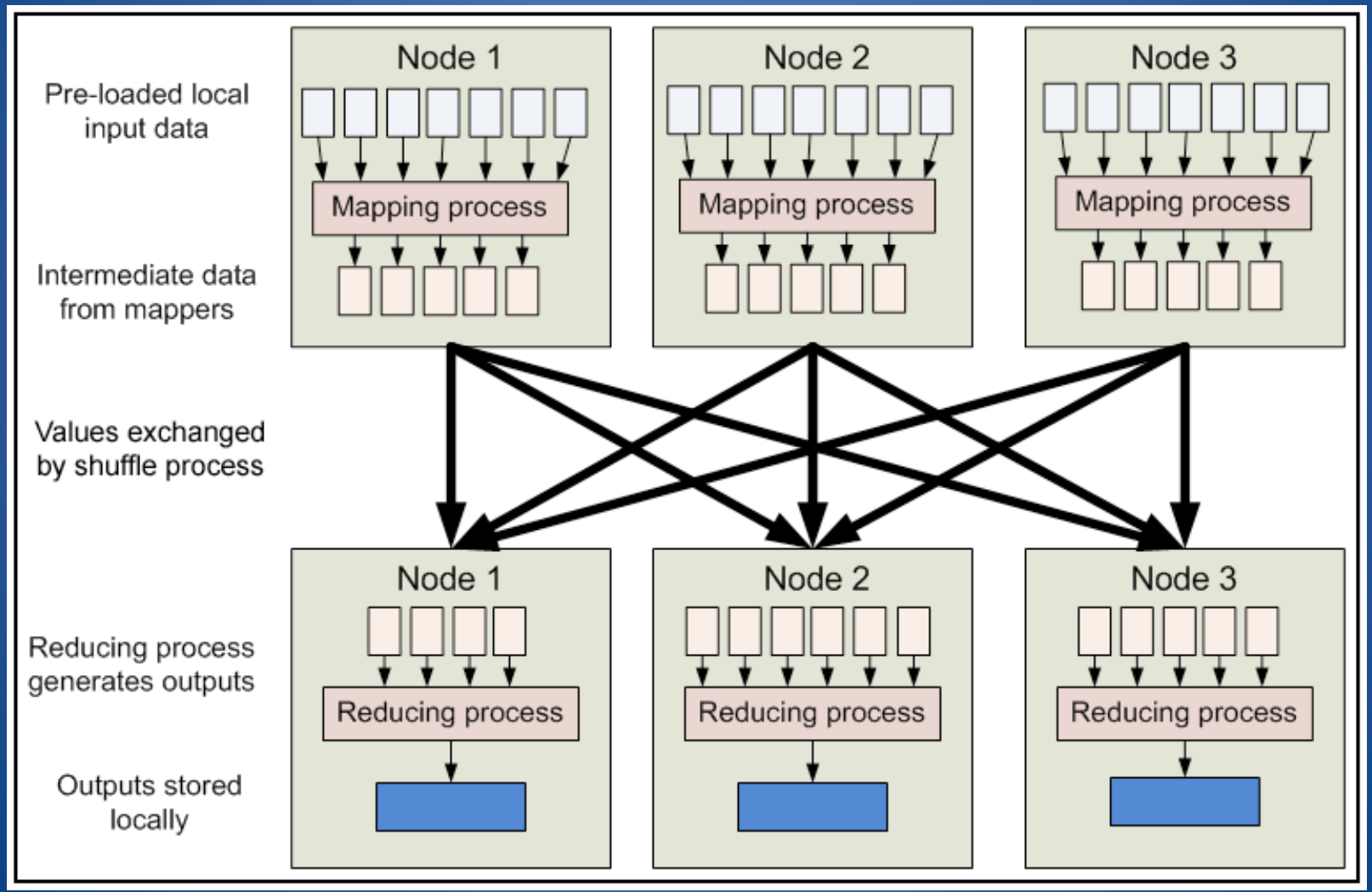
Reference Dirichlet Implementation

```
private void iterate(int iteration, DirichletState<Observation> state) {

  // create new posterior models
  Model<Observation>[] newModels = modelFactory.sampleFromPosterior(state
      .getModels());

  // iterate over the samples, assigning each to a model
  for (Observation x : sampleData) {
    // compute normalized vector of probabilities that x is described by each model
    Vector pi = normalizedProbabilities(state, x);
    // then pick one cluster by sampling a Multinomial distribution based upon them
    // see: http://en.wikipedia.org/wiki/Multinomial_distribution
    int k = UncommonDistributions.rMultinom(pi);
    // ask the selected model to observe the datum
    newModels[k].observe(x);
  }

  // update the state from the new models
  state.update(newModels);
}
```

Dirichlet Mapper on Hadoop

```
public void map(WritableComparable<?> key, Text value,
    OutputCollector<Text, Text> output, Reporter reporter) throws IOException {
  // read the next sample point
  Vector sample = DenseVector.decodeFormat(value.toString());
  // compute a vector of probabilities that sample is described by each model
  Vector pi = normalizedProbabilities(state, sample);
  // then pick one model by sampling a Multinomial distribution based upon them
  // see: http://en.wikipedia.org/wiki/Multinomial_distribution
  int k = UncommonDistributions.rMultinom(pi);
  // output value with key of selected model
  output.collect(new Text(String.valueOf(k)), value);
 }
```

Map/Reduce Jobs Use Local Data



Dirichlet Reducer on Hadoop

```
public void reduce(Text key, Iterator<Text> values,
    OutputCollector<Text, Text> output, Reporter reporter) throws IOException {
// load the model for this set of values
  Integer k = new Integer(key.toString());
  Model<Vector> model = newModels[k];
  while (values.hasNext()) {
    Vector v = DenseVector.decodeFormat(values.next().toString());
    // ask the selected model to observe the datum
   model.observe(v);
  }
 // compute & set new model parameters based upon the observations
  model.computeParameters();
  state.clusters.get(k).setModel(model);
  // output the cluster state for the next iteration
  output.collect(key, new Text(cluster.asFormatString()));
 }
```

Conclusion

- This is just the beginning
- High demand for scalable machine learning
- Contributors are needed who have
  - Interest, enthusiasm & programming ability
  - Test driven development skills
  - Comfort with the scary math (or bravery)
  - Interest and/or proficiency with Hadoop
  - Some large data sets you want to analyze

http://lucene.apache.org/mahout/