Jumping Into a Pool of Data

Aaron Craig, NZRS

p: (+64) (4) 931 6977
e: registry@nzrs.net.nz
f: (+64) (4) 931 6979
w: www.nzrs.net.nz
P.O. Box 24361, Wellington, New Zealand
New Zealand Domain Name Registry Limited

NZRS' Role

- We maintain 4/7 domain name servers for the .nz namespace
- Over 2.5 years of DNS data on a hadoop cluster
- Tools
  - Hive; Python, pandas, ipython, matplotlib; Java, Apache libraries

The Domain Name System

- You type in “google.co.nz.” and it gives you the IP address for Google
- Does other stuff as well
- Makes the internet work
- Computers exchange DNS queries

Diving into DNS

- Popularity tracking
  - Looking at DNS traffic over election year
  - Did some parties get more traffic during political events?
- Anomaly detection
  - Abuse of DNS
  - Particularly botnet activity
  - Can we program a computer to find this stuff automagically?

Botnets

- Group of infected computers
- Controlled by a bot-herder
- Bots are instructed to send spam or engage in DOS attacks
- Bots rendezvous at C&C servers to get those instructions
- Herder wants these meetings to slip under the radar

Botnets

- Problem
  - Big bursts of activity are suspicious
  - Domain gets blacklisted
  - Can't spam anymore :(
- Solution
  - Keep using a different domain
  - Need to register and deregister lots
  - Domain-generation algorithms

Domain-Generation Algorithms

- Come up with a valid domain name, register, wait, deregister
- Most names look meaningless
  - Conficker: tkggvtqvj.org
- Let's make a computer do this for us
- Aside: what is "meaningless"?

String Randomness

- Need a reference point
- Meaningless looking strings can have domain-specific meaning
  - ftp.website.org.nz
  - jzrs.net.nz
  - dgdgdgdgdgdgdg.linguist.nz
  - erogінglieurbgw.stuff.co.nz

String Distance Metrics

- We need to turn strings into numbers so they can be compared
- We can turn strings (one, or a group of them) into their *n*-gram distributions
- Probability distributions can then be compared

N-gram Distribution

- The unigrams of "kerikeri" are:
  - k -> 2/8
  - e -> 2/8
  - r -> 2/8
  - i -> 2/8
- The bigrams of "kerikeri" are:
  - ke -> 2/7
  - er -> 2/7
  - ri -> 2/7
  - ik -> 1/7

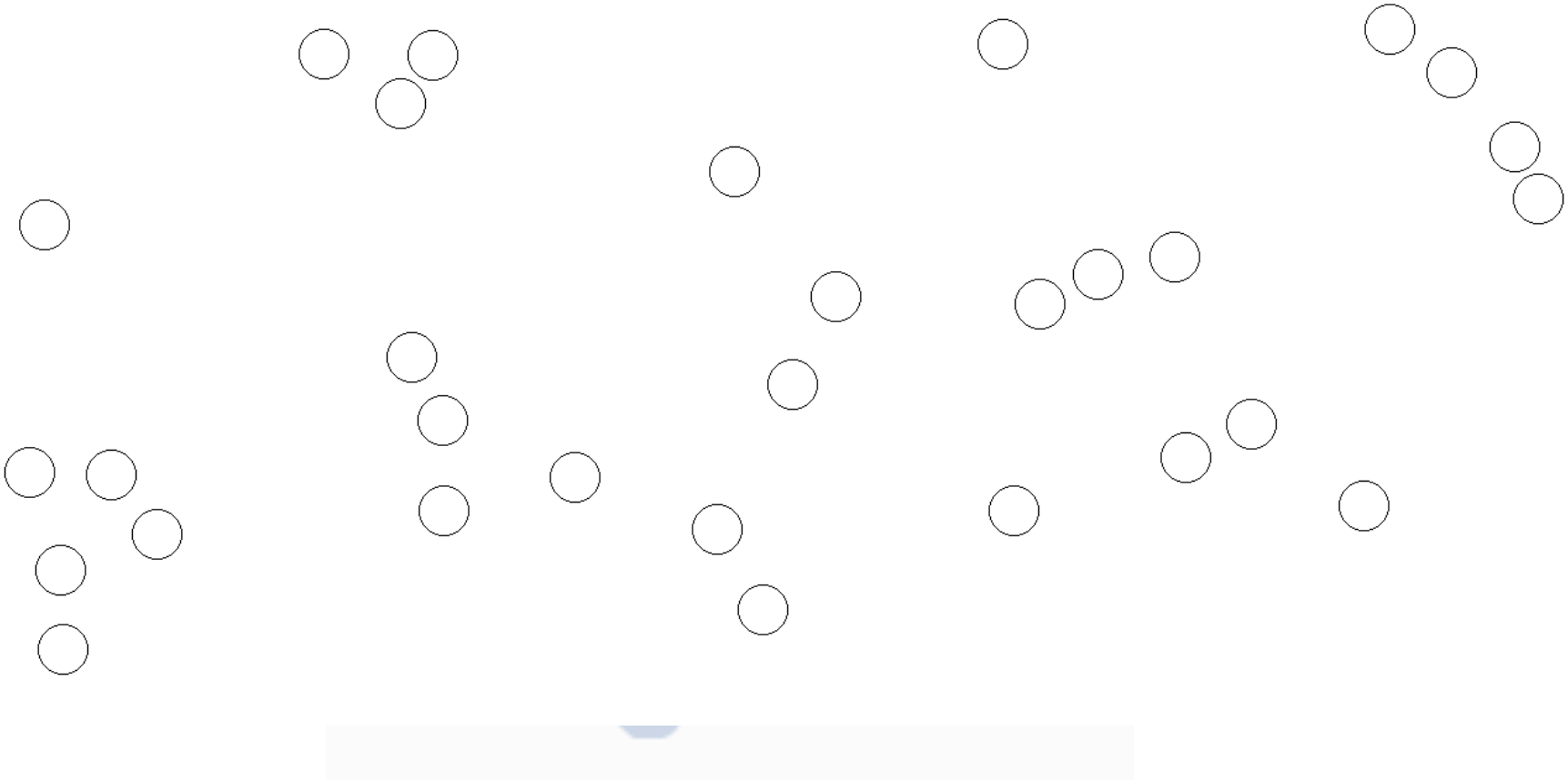
String Distance Metrics

- Comparing distributions:
  - Bhattacharyya
  - Kullback-Leibler
- Not that good if distribution represents one string
- Better if we cluster similar domains together to smooth outliers
- How do we cluster strings together?

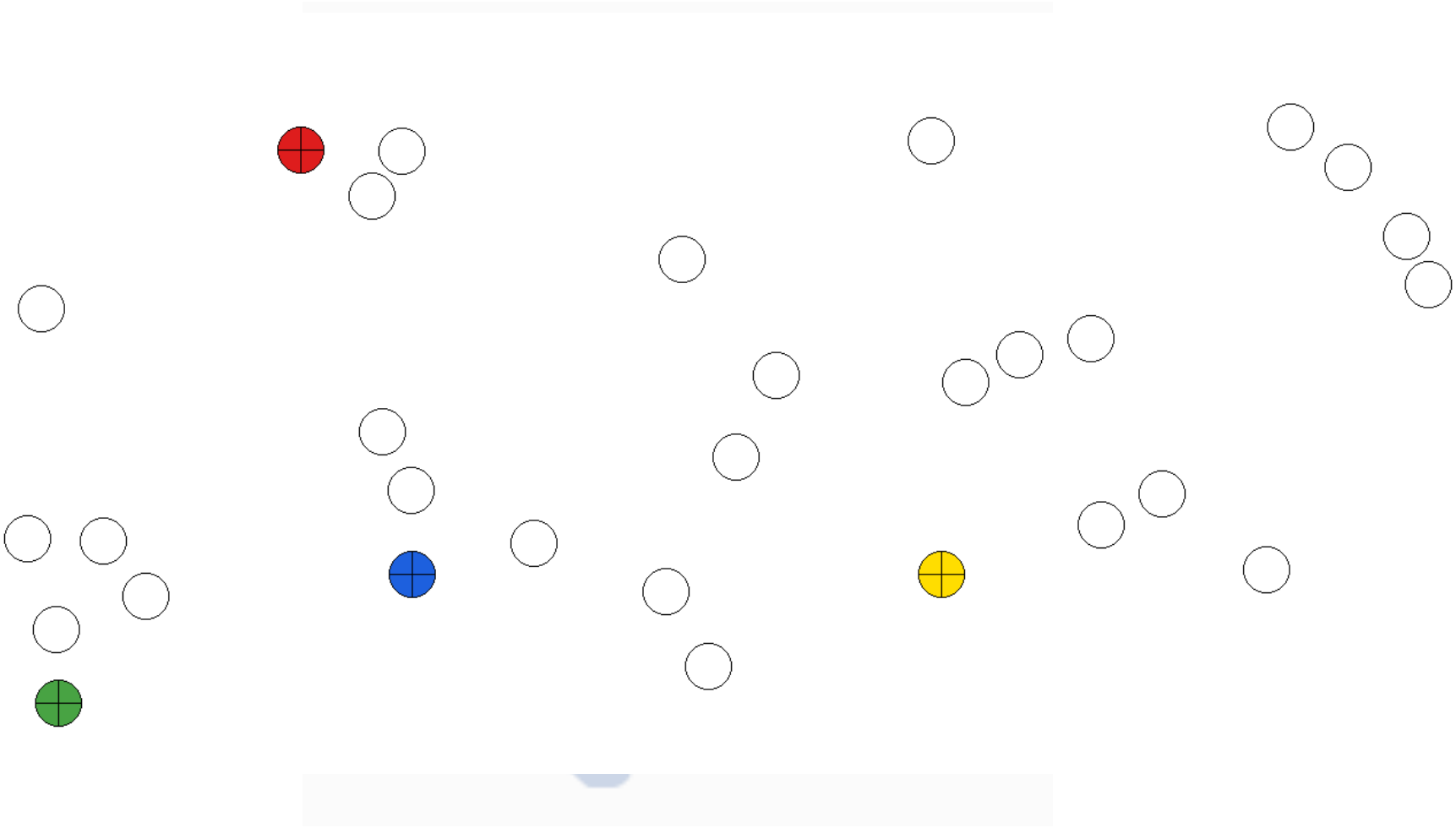
K-means Clustering

- Input: some points, an integer *k*
- The points grouped into *k* clusters
- The algorithm:
  - 1. Make *k* centroids, one for each cluster
  - 2. Assign each point to closest centroid
  - 3. Move centroids to the middle of the points in their clusters
  - 4. Repeat 2 and 3 until you're happy

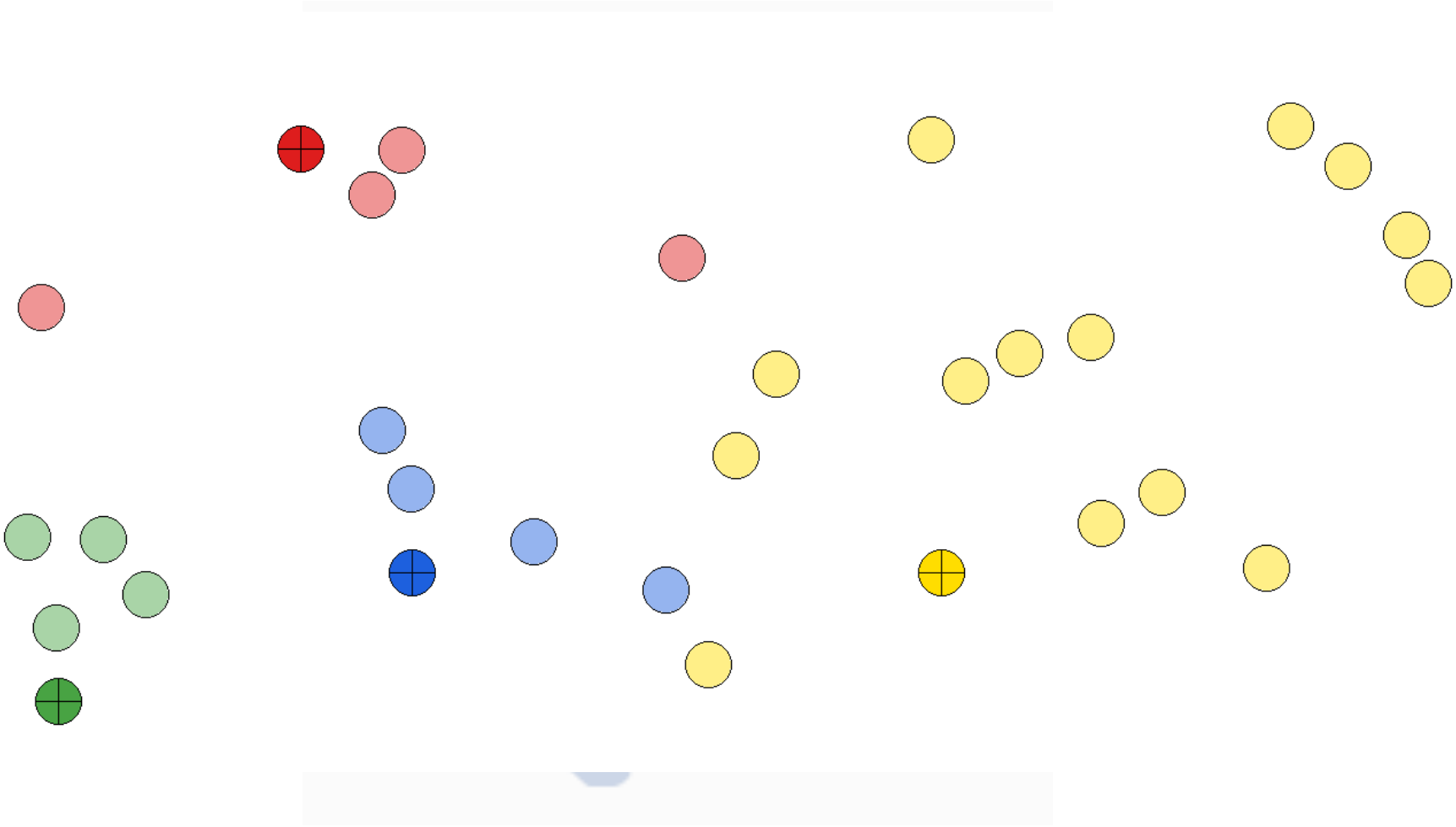
Here's a random world...



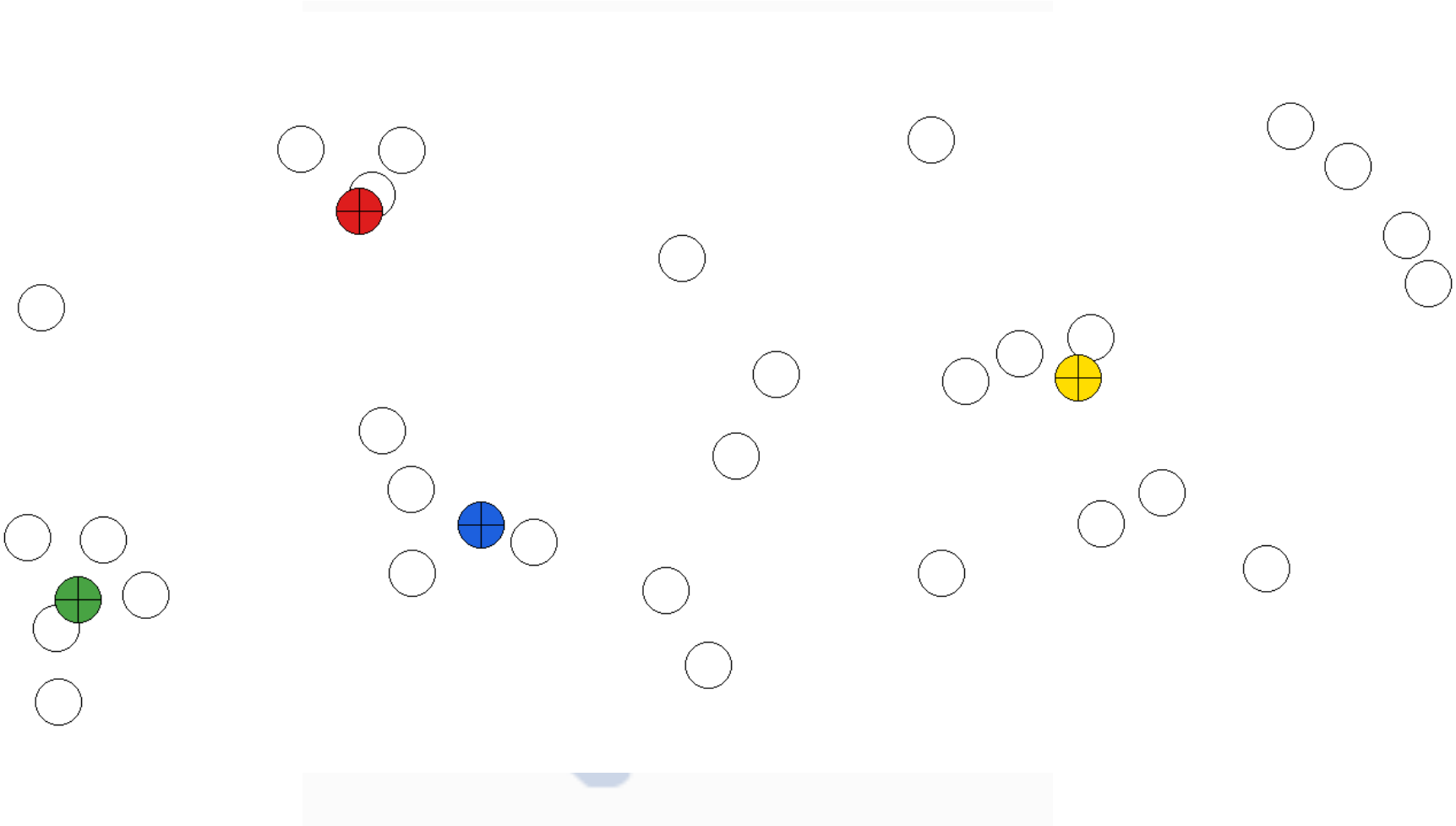
Let's do k=4. Place 4 centroids



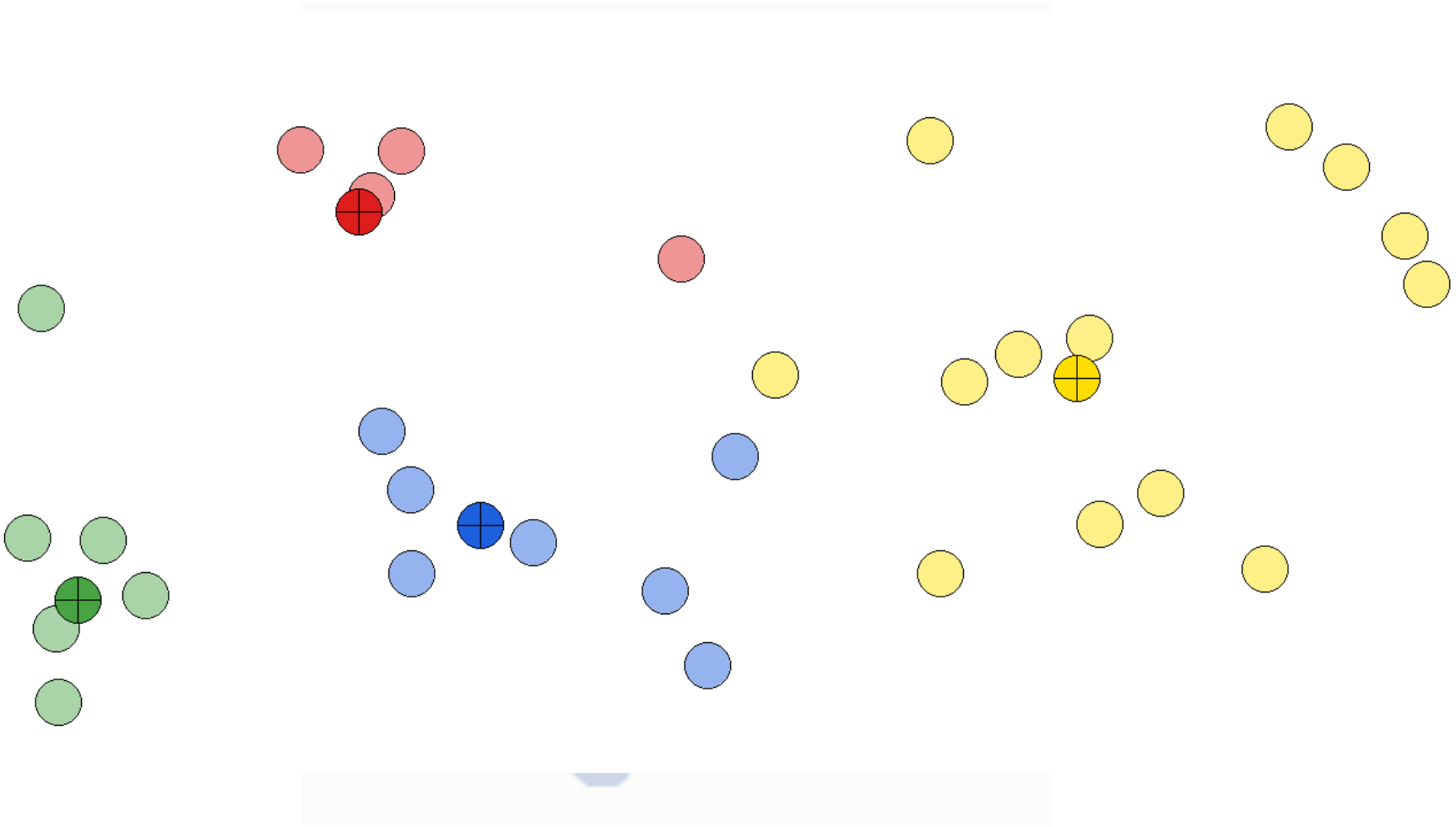
Assign all points to the closest centroid



Move each centroid to the middle of its cluster



Repeat until you're happy...



Clustering for Domains

- We clustered on vectors that represented a location in a 2D plane
- We want to cluster on vectors that represents domains queried
- Need “feature vectors” whose components are structural features about the domains they represent

Feature Vectors

- For each IP address partition the domains it queried into size *m* blocks
- Compute feature vectors for each block with components like:
  - Descriptive stats about n-grams
  - Average length of domain names
  - Shannon entropy of the domain name

Test Runs

- 676 good queries, 676 conficker queries
- Put into a big list and sort
- If clustering doesn't work each cluster should contain about 50% Conficker queries

Test Run

- Cluster 1: 262/532, 49% Conficker
- Cluster 2: 41/84, 49% Conficker
- Cluster 3: 45/62, 73% Conficker
- Cluster 4: 104/210, 50% Conficker
- Cluster 5: 224/464, 48% Conficker

Test Run

- Cluster 1: 85/213, 40% Conficker
- Cluster 2: 82/116, 71% Conficker
- Cluster 3: 302/626, 48% Conficker
- Cluster 4: 27/60, 45% Conficker
- Cluster 5: 180/337, 53% Conficker

Test Run

- Cluster 1: 69/153, 45% Conficker
- Cluster 2: 217/441, 49% Conficker
- Cluster 3: 208/347, 60% Conficker
- Cluster 4: 170/381, 45% Conficker
- Cluster 5: 12/30, 40% Conficker

Test Run

- Cluster 1: 105/223, 47% Conficker
- Cluster 2: 56/67, 84% Conficker
- Cluster 3: 280/563, 50% Conficker
- Cluster 4: 14/30, 47% Conficker
- Cluster 5: 221/469, 47% Conficker

Considerations

- Performs better if similar domains are grouped into the same vectors
- Sensitive to initial choice of centroids
- Need to specify number of clusters but there are ways around this
- After clustering, we can use our distribution comparison metrics

The Dream

- Experiment with the components of the feature vectors
- Repeat tests with some clusters being at least 80% Conficker
- Refine, run on live data
- Cluster by host-domain graph
- Be able to tell which botnet a cluster belongs to

Shout Outs

- “Detecting the Rise of DGA-Based Malware” Antonakakis et al
- “Detecting networks Employing Algorithmically Generated Domain Names” Ashwath Kumar Krishna Reddy
- Seb, Jamie, Daniel