

Computer Intrusion Detection  
Using Features From Graph  
Theory and Algebraic Topology

Michael Postol

Tom Goldring

U.S. National Security Agency

July 2004

# Outline

- I. User Profiling
- II. Motivation
- III. Process and Window Graphs
- IV. Classification and Regression Trees
- V. Random Forests
- VI. Initial Experiments and Results
- VII. Interval Graphs
- VIII. Features From Algebraic Topology
- IX. Conclusions and Future Directions

# User Profiling

- Intrusion can involve
  - Illegally obtained passwords.
  - Machines left unattended.
  - Malicious use by an authorized user. (Insider Threat)

# User Profiling

- Goal: Distinguish authorized users from impostors.



# Process and Window Graphs

- Data consists of 30 sessions each of 10 users on WINDOWS NT machines.
- Data for each session represented by 2 directed graphs.

# Process and Window Graphs

- Process graphs have vertices representing the processes called by the user and a directed edge from process 1 to process 2 if process 2 is spawned by process 1.
- Process graphs are disconnected trees since processes called by the operating system are omitted.

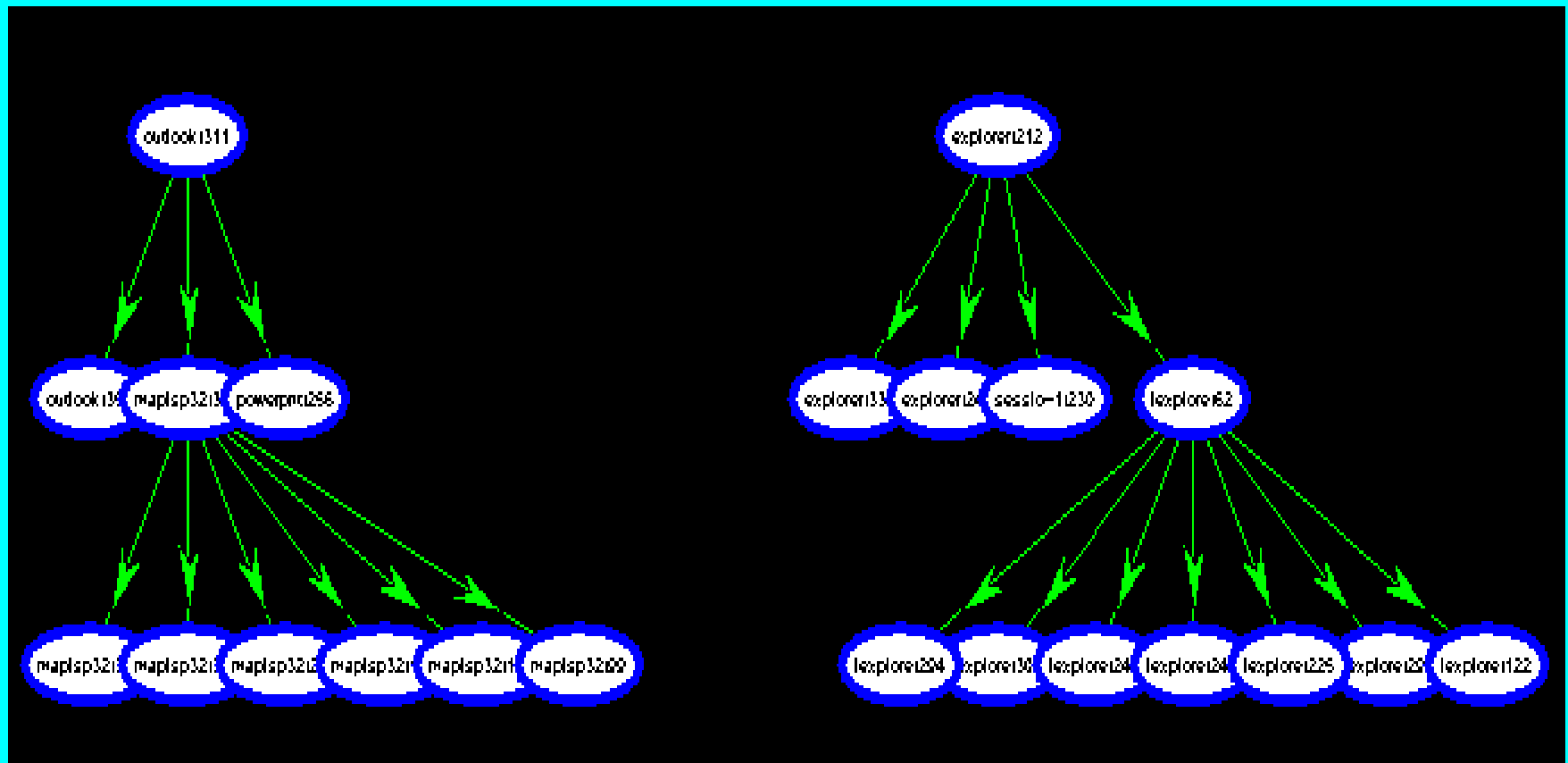
# Process and Window Graphs

- Window graphs have vertices representing windows clicked on by user with a directed edge from a window to the next one clicked on.
- Window graphs are connected and can have cycles.

# Process and Window Graphs

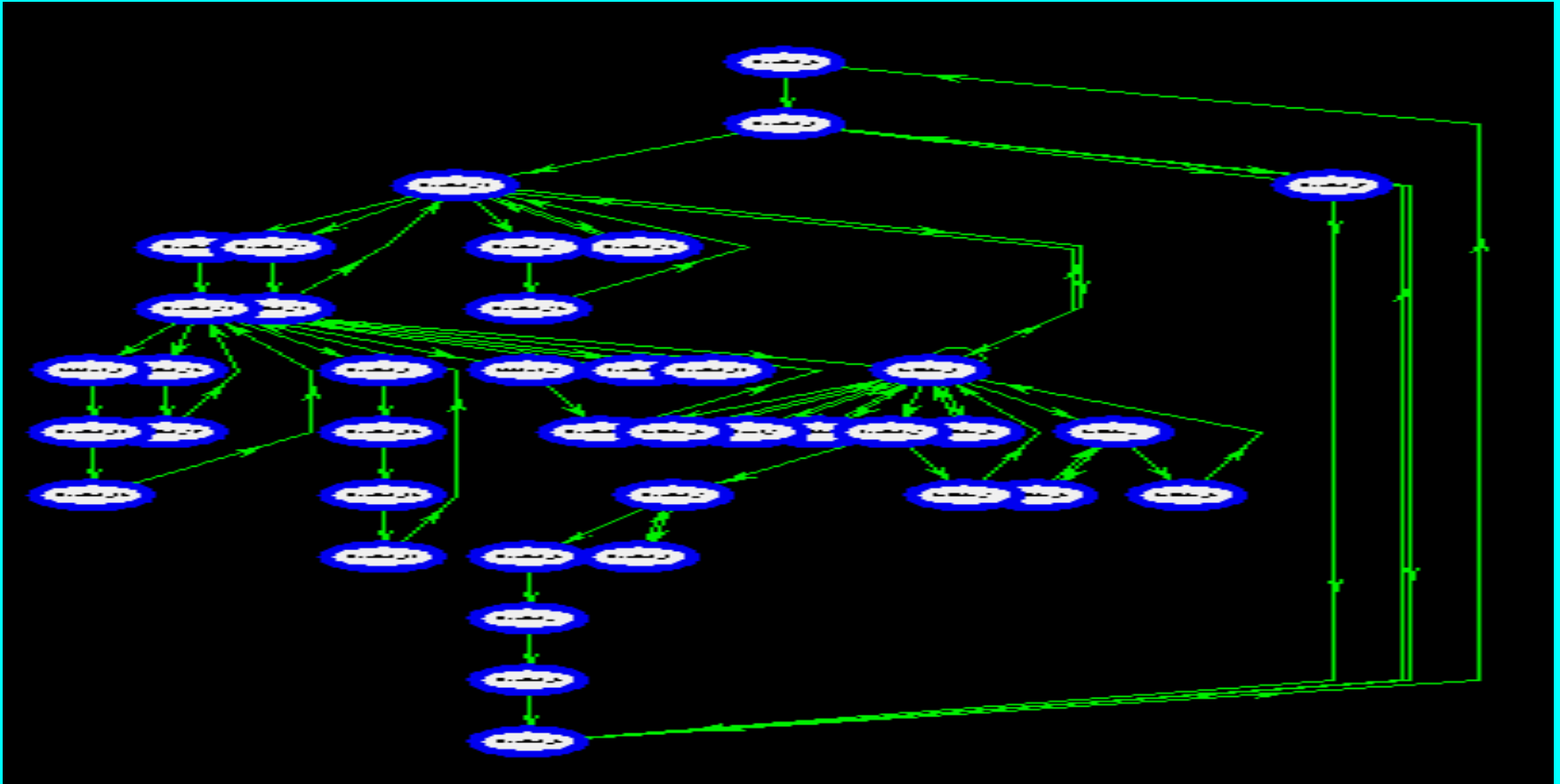
- Both types of graphs have edge weights representing time (in seconds) between the start of processes (process graphs) or the time between first clicks on consecutive windows (window graphs).

# Process and Window Graphs



Example-Process Graph

# Process and Window Graphs



Window Graph-Example

# Classification and Regression Trees

- We used “Random Forests” to test for each user vs. someone else.
- Based on Classification and Regression Trees (CART).
- For each sample associate a vector of  $N$  features.
- We want to classify samples into  $J$  classes.

# Classification and Regression Trees

- In our case  $J=2$ .
- At each node in tree we split region in  $N$ -space along one of the features.
- We split in a manner which causes the largest decrease in an “impurity function”  $Q$ .
- If  $t$  is a node in the tree, let  $R(t)$  be the corresponding region in space.

# Classification and Regression Trees

- We use the “Gini function”

$$Q(t) = 1 - \sum_j [p(j | t)]^2$$

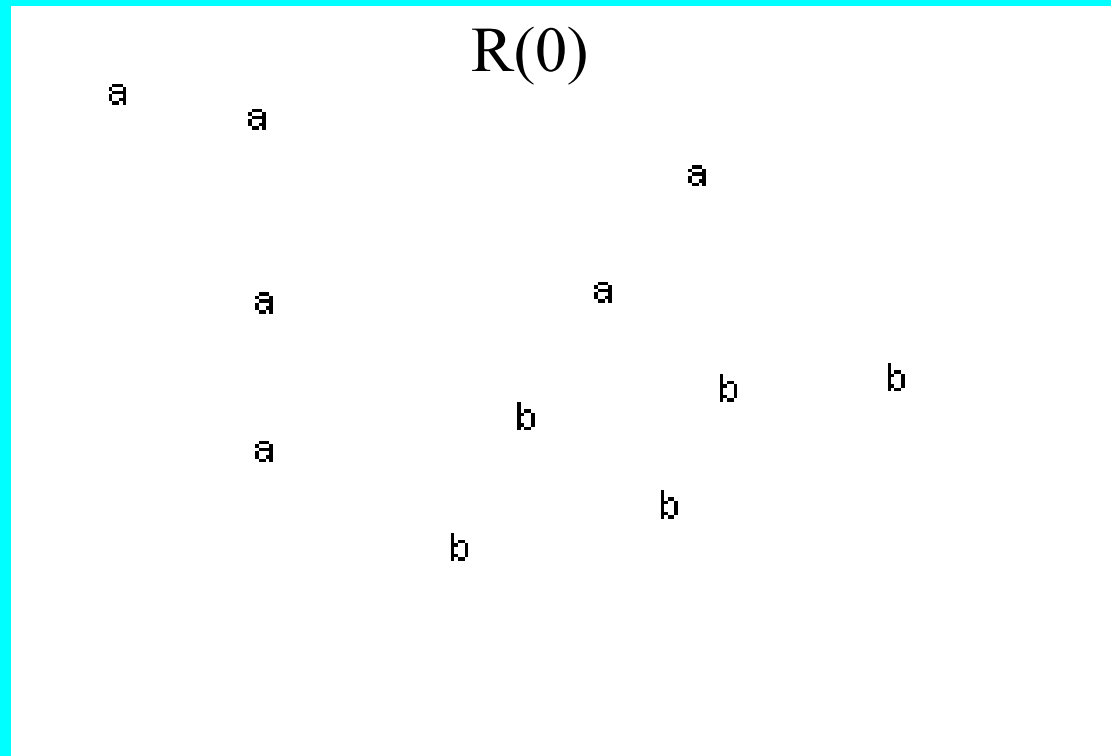
- where  $p(j|t)$  is the proportion of class  $j$  in  $R(t)$ .
- This ranges from 0 if all samples in the region are of the same class to  $1/J$  if there are an equal number of samples from each class.

# Classification and Regression Trees

- Continue splitting until misclassification rate falls below a predetermined threshold.

# Classification and Regression Trees

$$Q(0) = 1 - (6/11)^2 - (5/11)^2 = 60/121$$

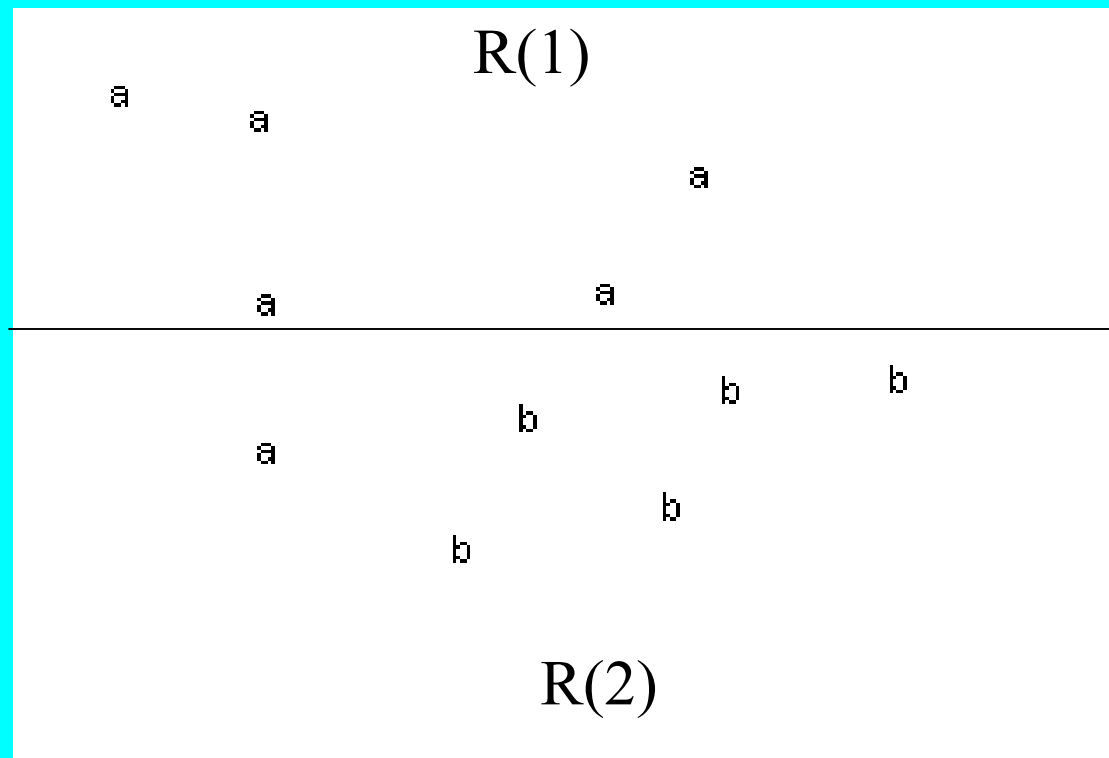


# Classification and Regression Trees

$$Q(0) = 1 - (6/11)^2 - (5/11)^2 = 60/121$$

$$Q(1) = 0$$

$$Q(2) = 1 - (1/6)^2 - (5/6)^2 = 10/36$$



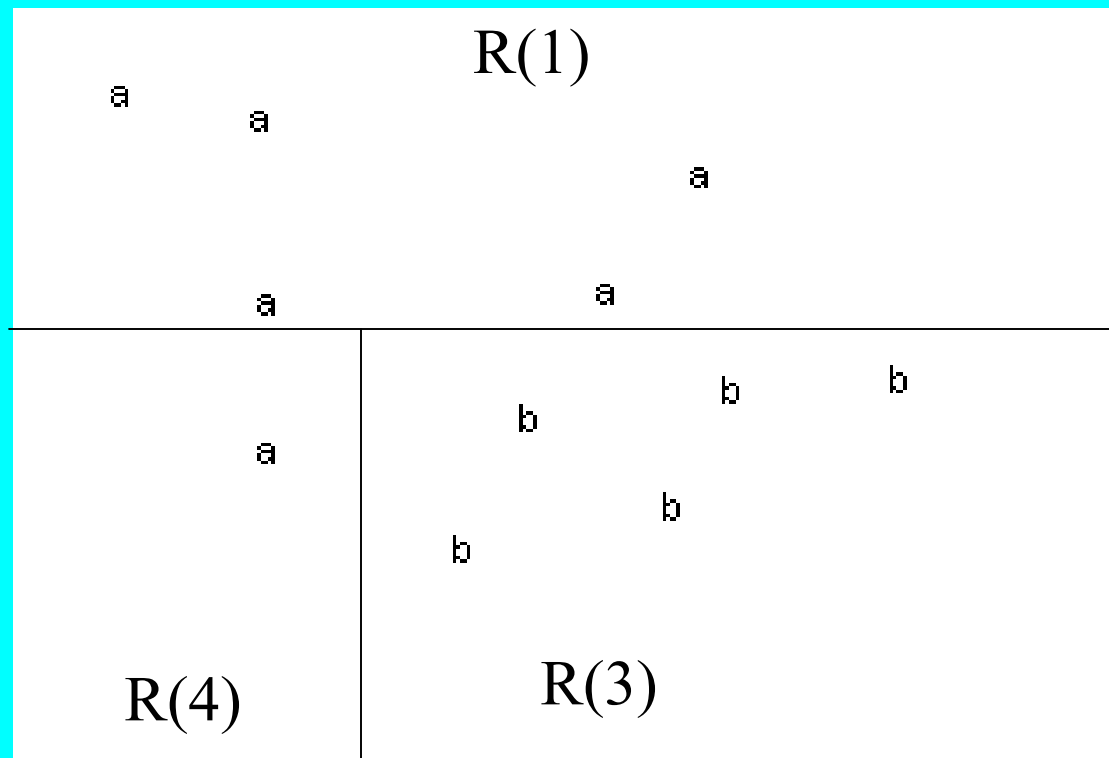
# Classification and Regression Trees

$$Q(0) = 1 - (6/11)^2 - (5/11)^2 = 60/121$$

$$Q(1) = 0$$

$$Q(2) = 1 - (1/6)^2 - (5/6)^2 = 10/36$$

$$Q(3) = Q(4) = 0$$



# Random Forests

- For random forests grow about 100 trees.
- For each tree, use a different subset of the features at each node. (about the square root of the total number)
- Randomly pick different training set for each tree.
- Train on bag of  $2/3$  of samples.

# Random Forests

- Test set is 1/3 of samples left “out of the bag”.
- Decisions made by majority vote over all trees.
- Classification can be binary or multi-way.
- Can also determine most important features by randomly perturbing them one at a time.

# Experiment

- Data: 10 users, 30 sessions each.
- 56-long feature vectors for each session.
- Features related to size and shape of window and process graphs as well as edge weights representing times between clicks on windows or the start of processes.
- 20 sessions for each user used to train 10 random forest models.

# Experiment

- Model  $j$  represents a decision on whether a session is user  $j$  or someone else.
- At this step we found the most important features.
- Window graphs more important than process graphs.

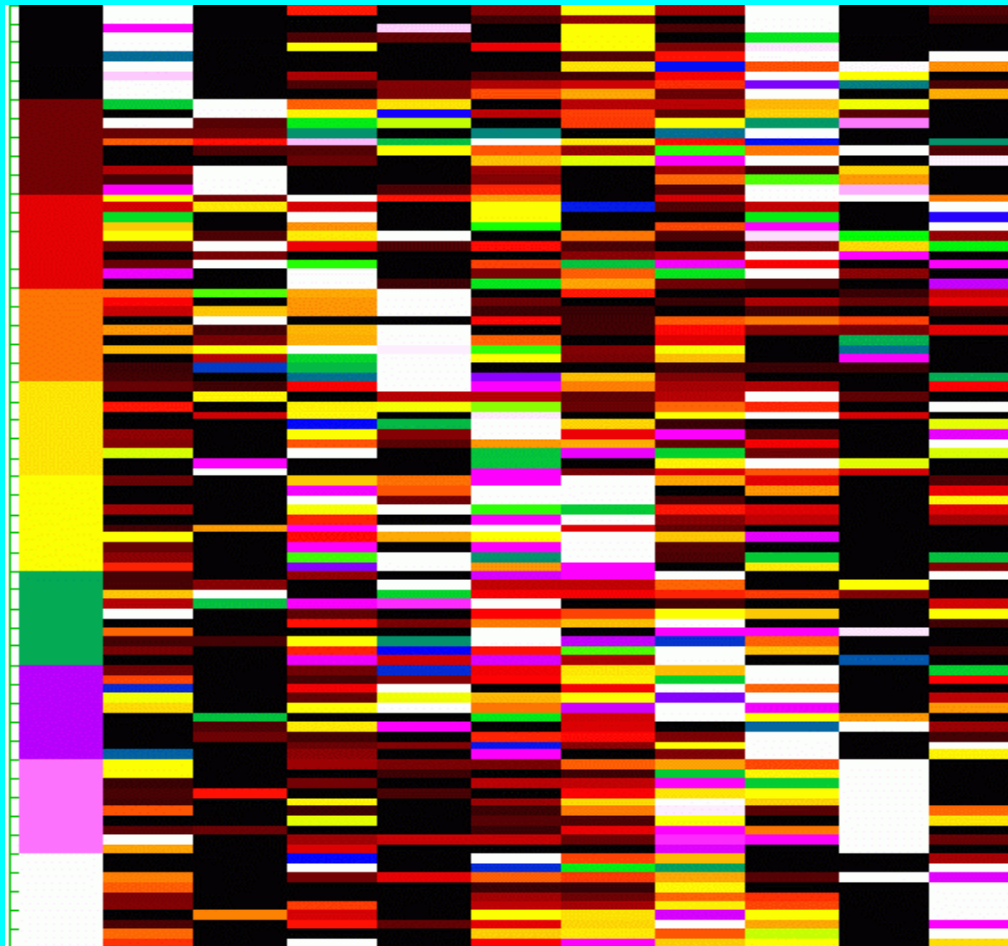
# Experiment

- Most important window graph features:
  - Sources-Did user return to first window used?
  - Sinks-Was last window used one which was used previously?
  - Timing factors-Total edge weight and total weighted degree.

# Experiments

- For the 10 test sessions for each user, we ran them against all 10 random forest models.
- We obtain a 10 long vector for each session.
- Component  $j$  is a number between 0 and 1 representing the percentage of trees in model  $j$  voting for the session to belong to user  $j$  as opposed to someone else.
- Arrange these in a 10 X 10 matrix.

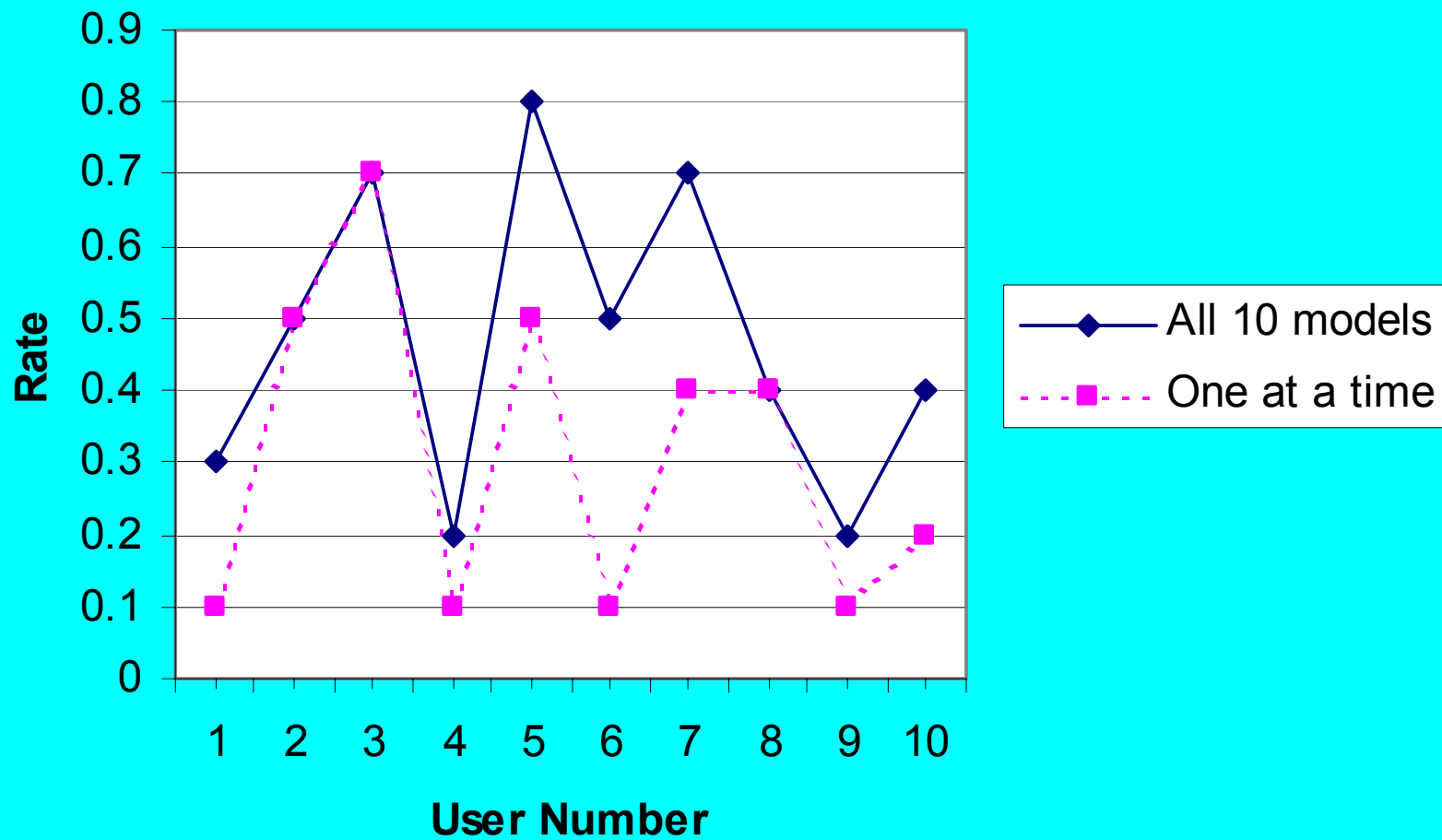
# Data Image Plot



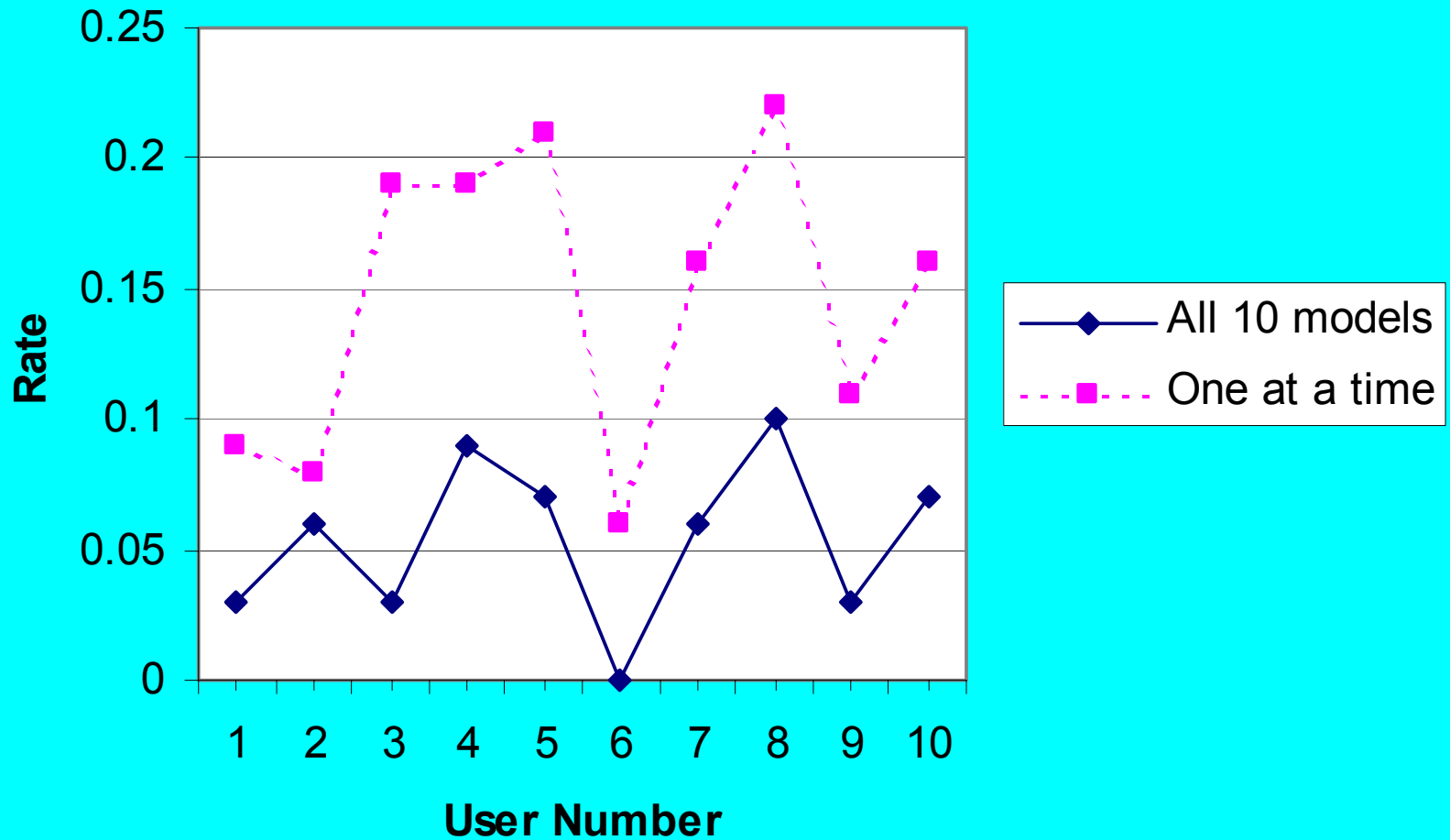
# Confusion Matrix

Truth:	1	2	3	4	5	6	7	8	9	10
1	7	1					1		1	
2		5	2	1	1		1			
3			3	1	1					1
4			1	8	1	4	1	1		
5		1			2	1	3			1
6						5				
7							3	2	1	1
8	2	2	2		3			6		
9		1						1	8	1
10	1		2		2		1			6

## False Alarm Rate



## False Authentication Rate



# Results

- Looking at the forests one at a time instead of all together invariably lowered the false alarm rate and raised the false authentication rate.

# Interval Graphs

- Vertices represent windows.
- Edge between 2 windows up at the same time and directed from window open earlier to window open later.
- Advantages:
  - Distinguishes users who open many windows at once from neater users.
  - Graph has no directed cycles allowing for construction of poset complex described below.

# Simplicial Complexes

- Graphs give rise to some natural simplicial complexes:
  - The graph itself.
  - Complete Subgraph Complex
    - Vertices are graph edges and maximal simplices are edge sets forming a complete subgraph.
  - Neighborhood Complex
    - Vertices are graph vertices and maximal simplices consist of the vertex along with its neighbors.

# Simplicial Complexes

- Poset Complex
  - Vertices are graph vertices and maximal simplices are maximal directed paths.
- Matching Complex
  - Vertices are graph edges and simplices are edge sets forming a matching (I.e. no 2 edges are adjacent)

# Simplicial Complexes

- Broken Circuit Complex
  - Vertices are graph edges and maximal simplices are maximal connected edge sets containing no circuit (spanning trees for a connected graph).
  - Such edge sets form the independent sets of a matroid (a generalization of linear independence) and have particularly simple homology.

# Simplicial Complex Features

- Some potential features:
  - Dimension of complex.
  - Number of simplices in each dimension.
  - Euler characteristic. (An alternating sum of the number of faces in each direction)
  - Betti Numbers of Homology Groups
  - Cup Products and Steenrod Squares (Computed using Gonzalez-Diaz/Real algorithms.)

# Conclusions

- Random Forests show definite promise.
- Huge number of possibilities for new features.
- Need to determine which ones are meaningful for our problem.

# Future

- Work on coding these features.
- Find ways to deal with complexity-a problem even for small graphs.
- Find ways to incorporate more of the data into graphs and resulting feature vectors.
- Create larger data sets.