SVM Learning of IP Address Structure for Latency Prediction

Rob Beverly, Karen Sollins and Arthur Berger

{rbeverly,sollins,awberger}@csail.mit.edu

SVM Learning of IP Address Structure for Latency Prediction

- The case for Latency Prediction
- The case for Machine Learning
- Data and Methodology
- Results
- Going Forward

The Case for Latency Prediction

Latency Prediction (again?)

- Significant prior work:
  - King [Gummandi 2002]
  - Vivaldi [Dabek 2004]
  - Meridian [Wong 2005]
  - Others… IDMaps, GNP, etc…
- Prior Methods:
  - Active Queries
  - Synthetic Coordinate Systems
  - Landmarks
- Our work seeks to provide an <u>agent-centric</u> (single-node) alternative

Why Predict Latency?

1. *Service Selection*: balance load, optimize performance, P2P replication
2. *User-directed Routing*: e.g. IPv6 with per-provider logical interfaces
3. *Resource Scheduling*: Grid computing, etc.
4. *Network Inference*: Measure additional topological network properties

An Agent-Centric Approach

- **Hypothesis**: Two hosts within same sub network likely have consistent congestion and latency
- Registry allocation policies give network structure – but fragmented and discontinuous
- Formulate as a supervised learning problem
- Given latencies to a set of (random) destinations as training:
  - `predict_latency(unseen IP)`
  - `error = |predict(IP) – actual(IP)|`

The Case for Machine Learning

Why Machine Learning?

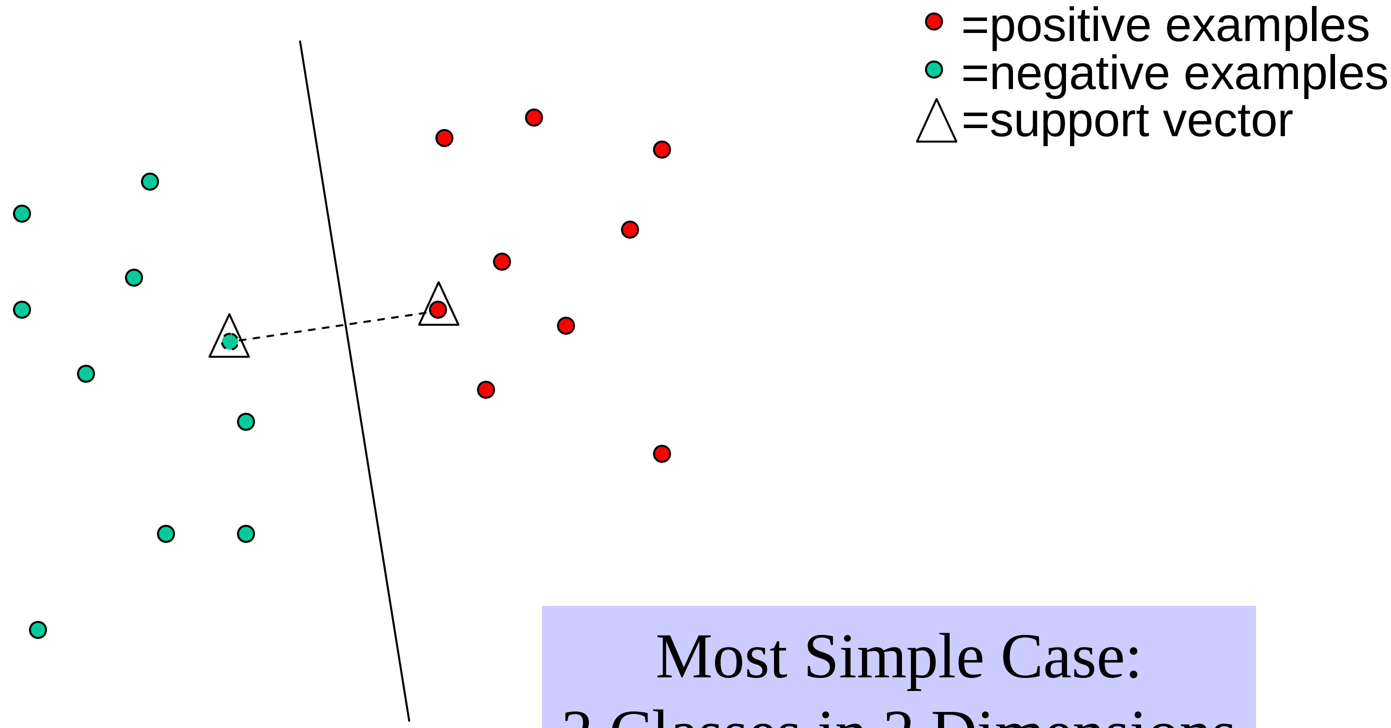
Internet-scale Networks:
- Complex (high-dimensional)
- Dynamic

- Can accommodate and recover from infrequent errors in probabilistic world
- Traffic provides large and continuous training base

Candidate Tool: Support Vector Machine

- Supervised learning (but amenable to online learning)
- Separate training set into two classes in most general way
- **Main insight**: find hyper-plane separator that maximizes the minimum margin between convex hulls of classes
- **Second insight**: if data is not linearly separable, take to higher dimension
- **Result**: generalizes well, fast, accommodate unknown data structure

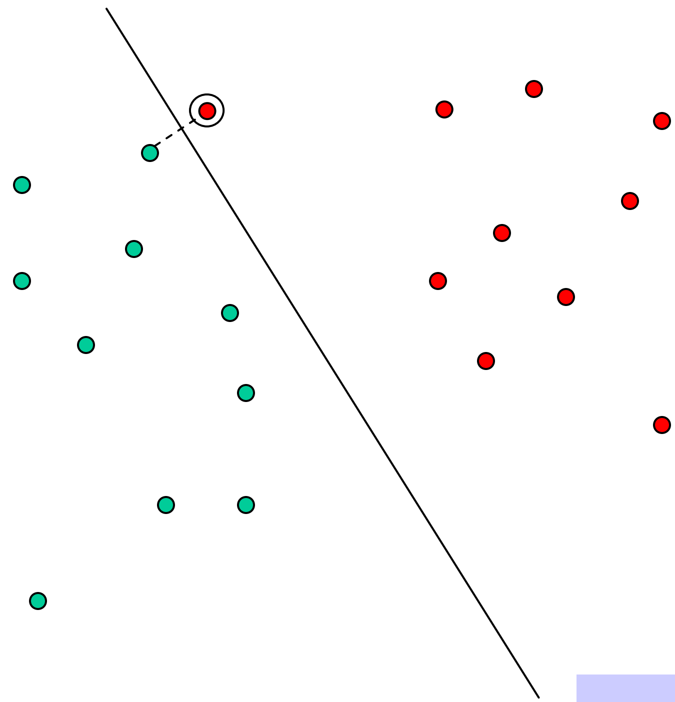
SVMs – Maximize Minimum Margin



Most Simple Case:
2 Classes in 2 Dimensions
Linearly Separable

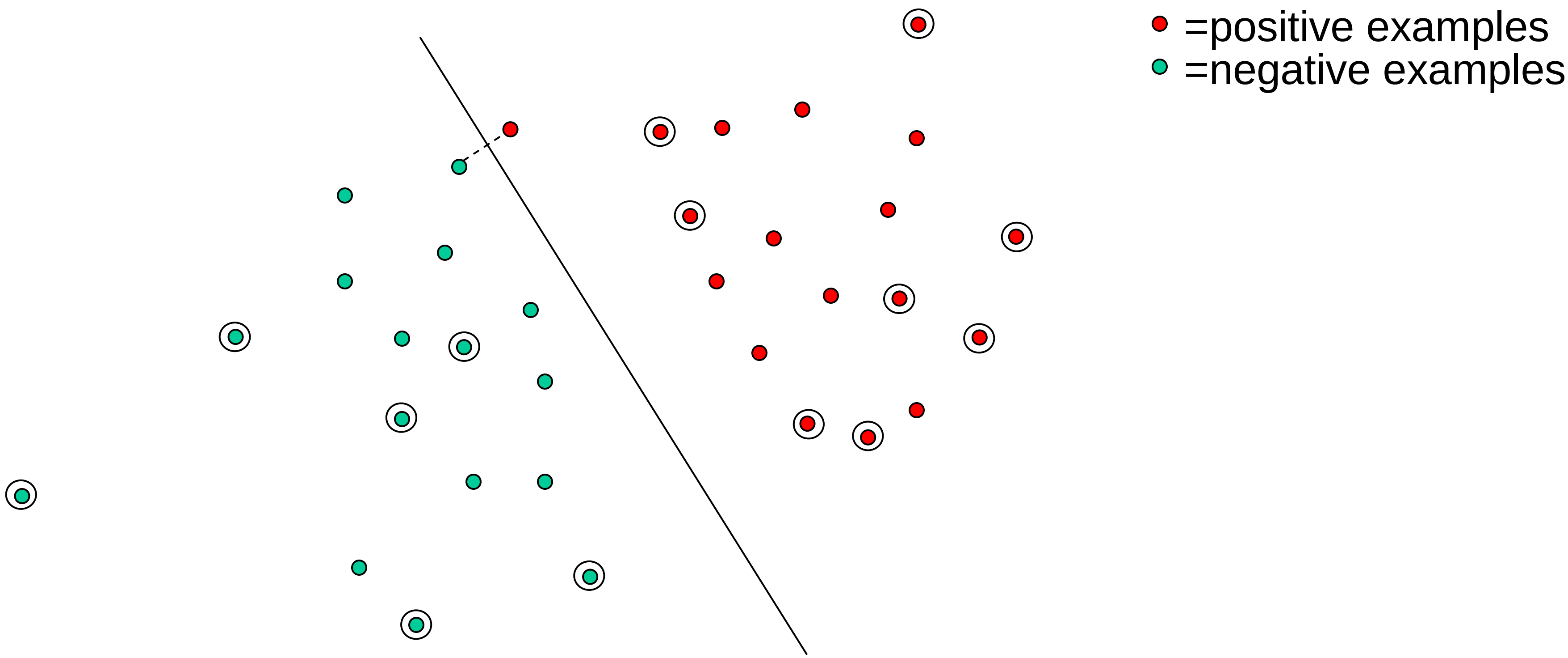
SVMs – Redefining the Margin

● =positive examples
● =negative examples

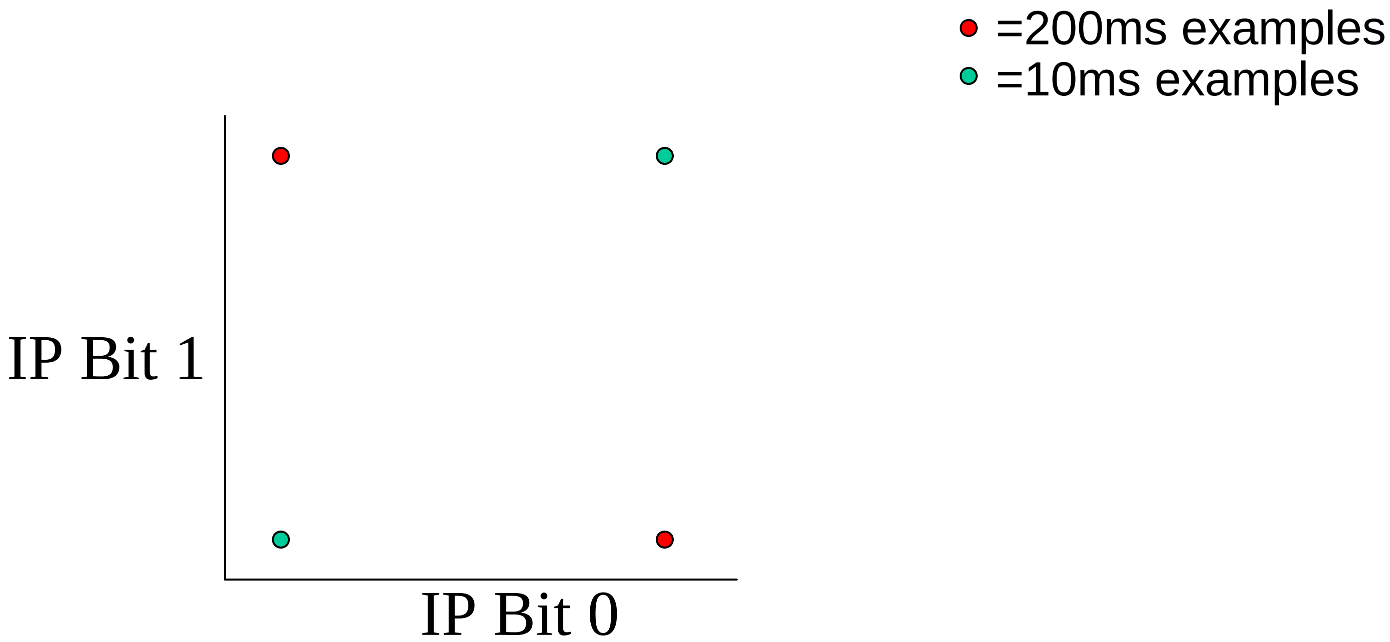


The single new positive example ⊙ redefines margin

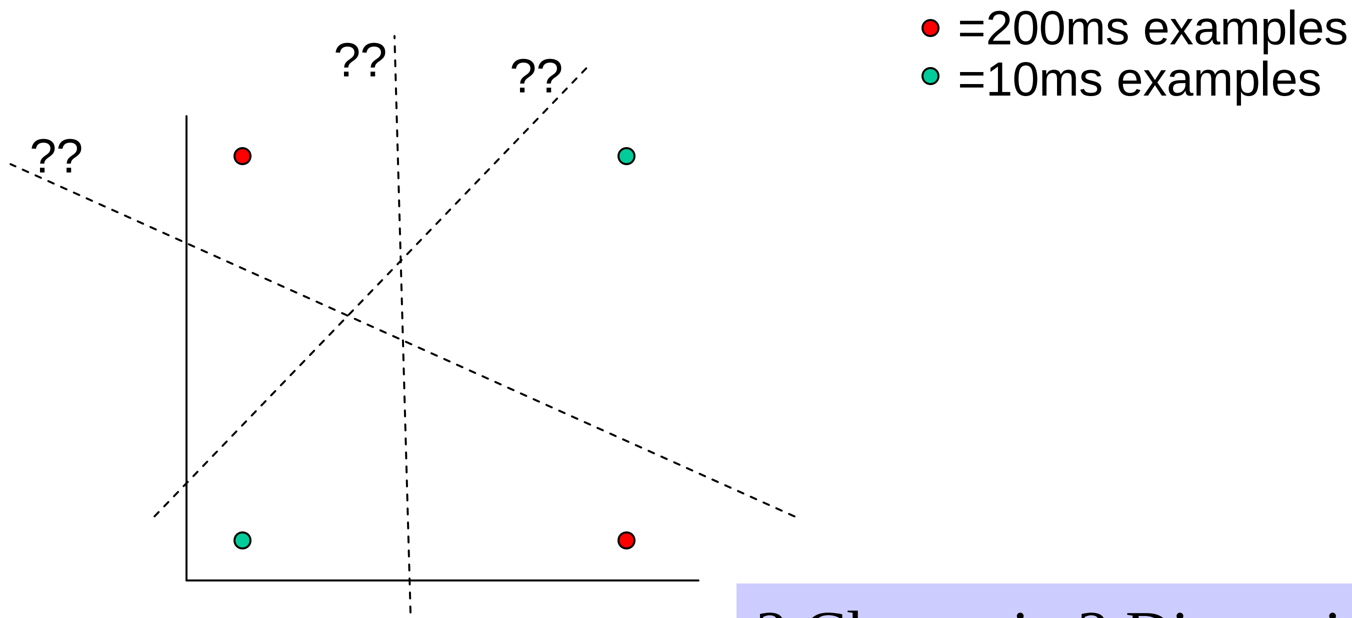
Non-SV Points don't affect solution



IP Latency Non-Linearity



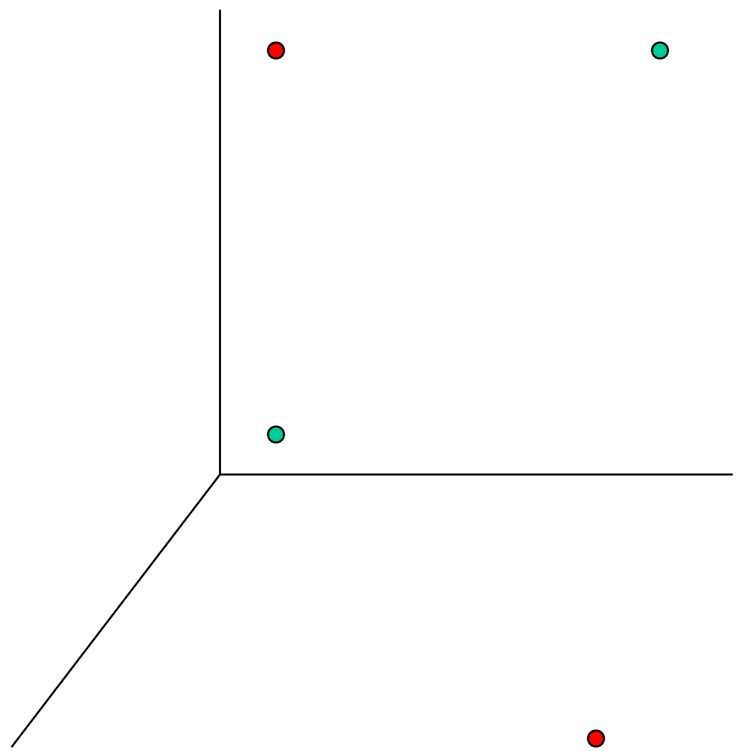
Higher Dimensions for Non-Linearity



2 Classes in 2 Dimensions
NOT Linearly Separable

Kernel Function Φ

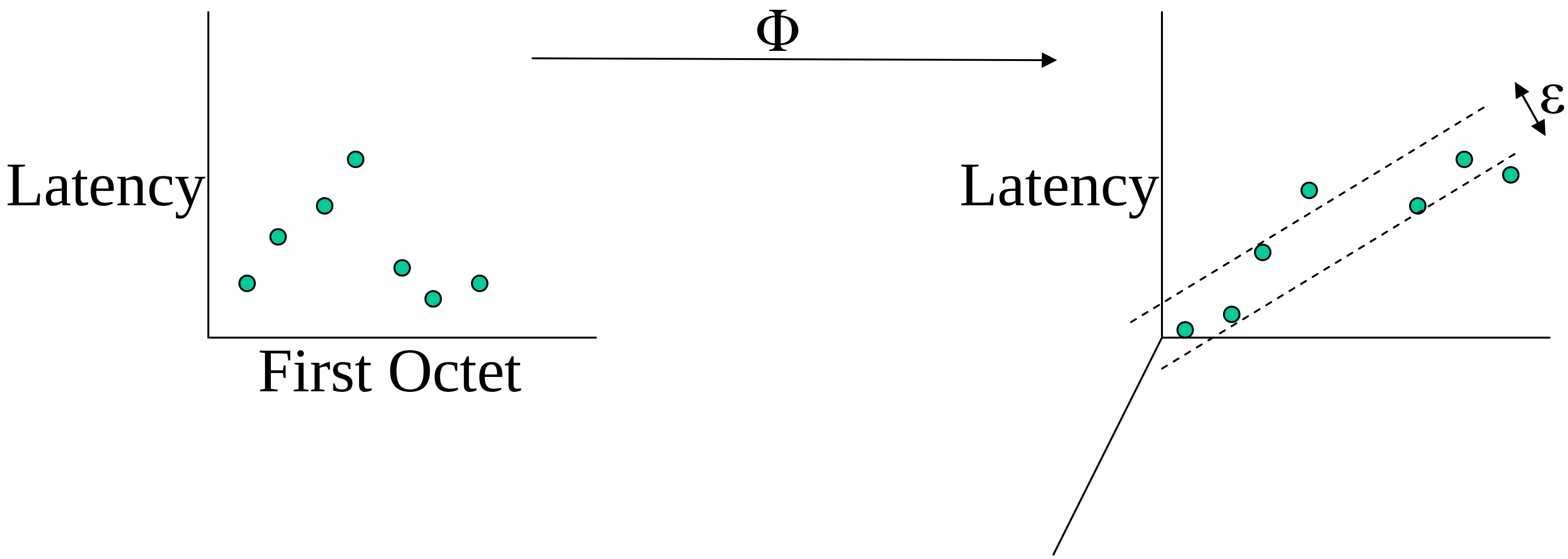
● =200ms examples
● =10ms examples



2 Classes in 3 Dimensions
Linearly Separable

Support Vector Regression

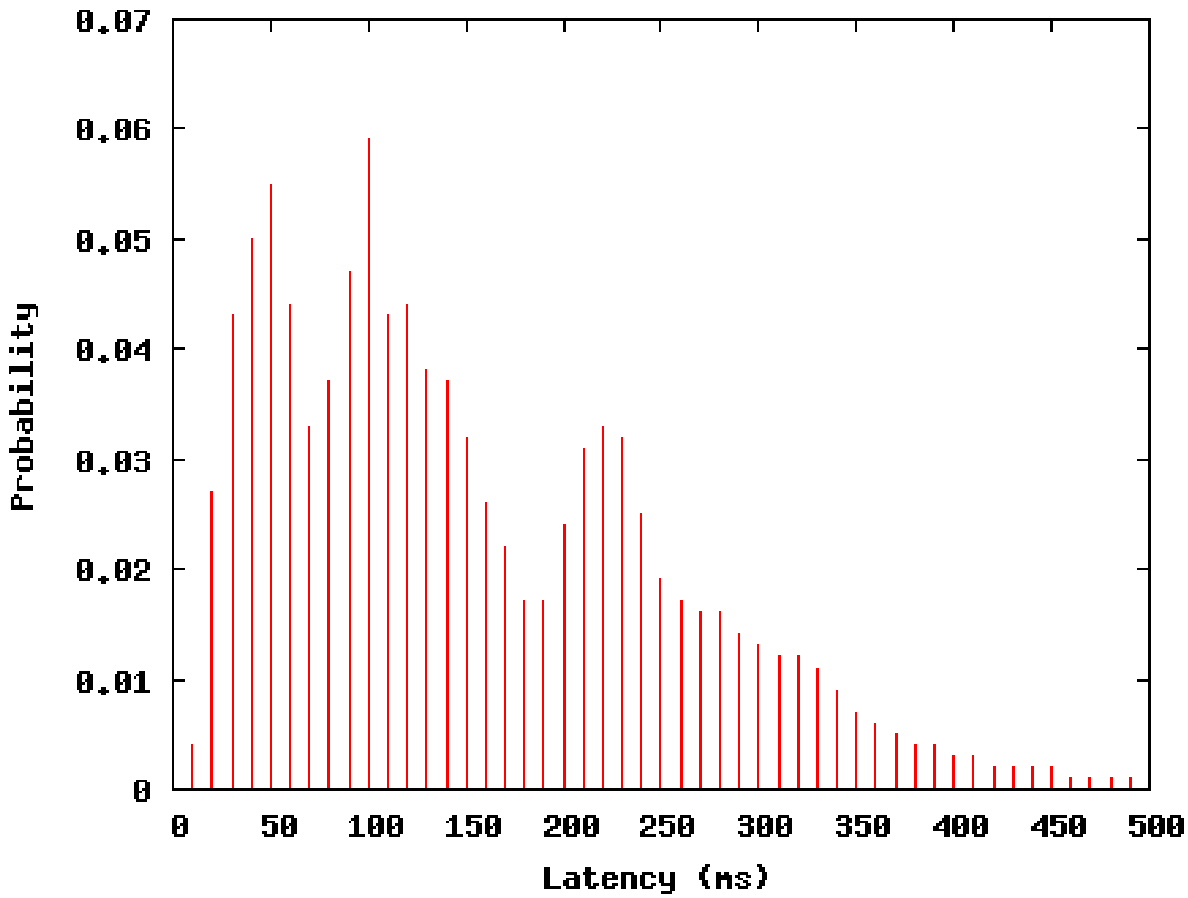
- Same idea as classification
- ε-insensitive loss function



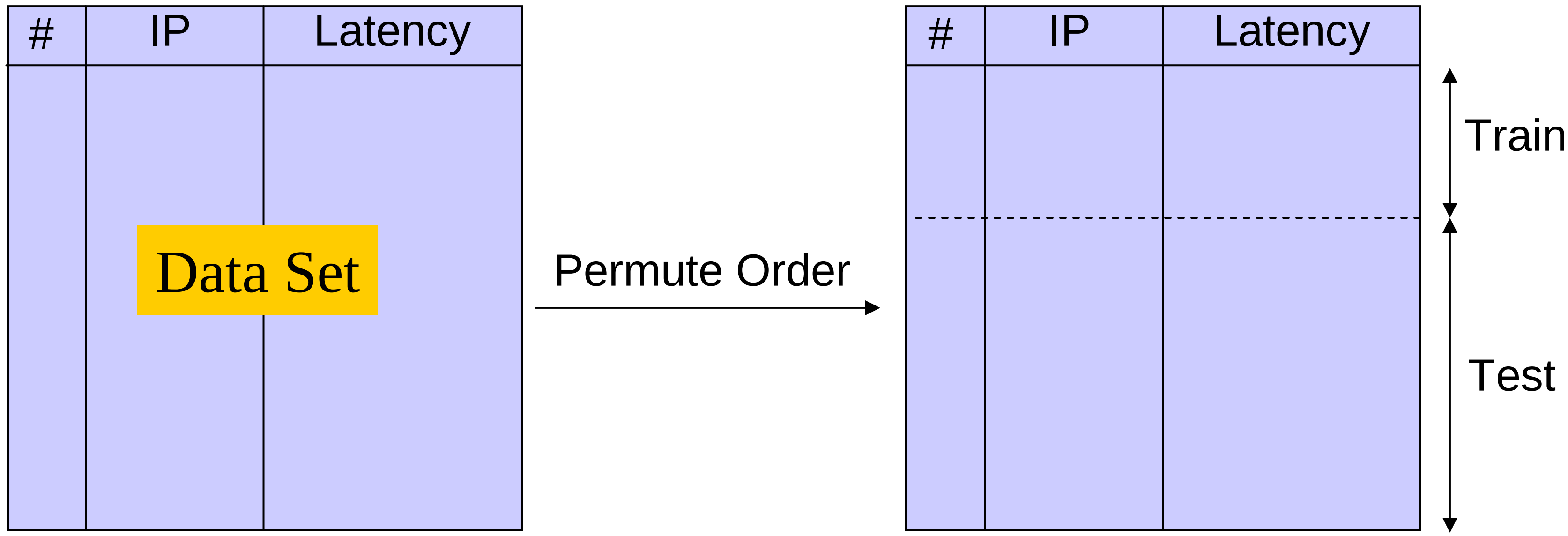
Data and Methodology

Data Set

- 30,000 random hosts responding to ping
- Average latency to each over 5 pings
- Non-trivial distribution for learning

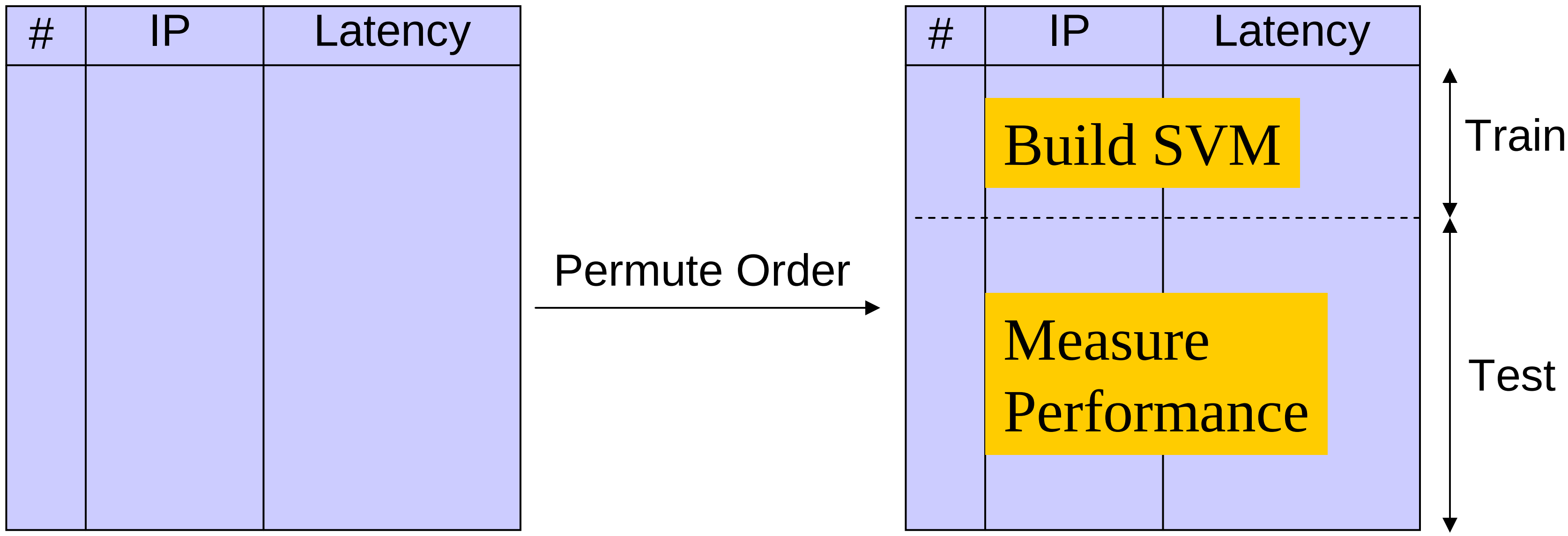


Methodology



- Average 5 experiments:
  - Randomly permute data set
  - Split data set into training / test points

Methodology



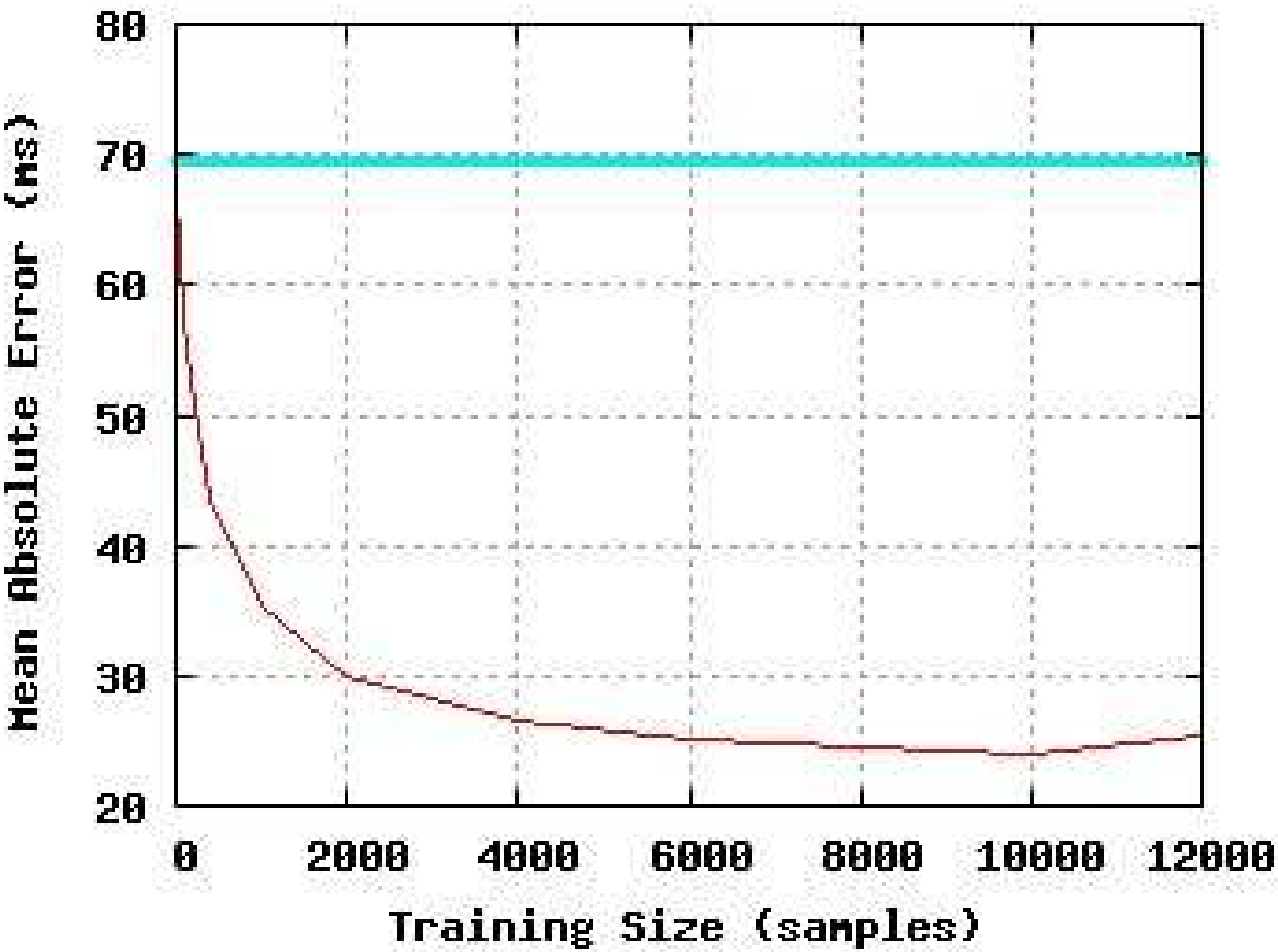
- Average 5 experiments:
  - Training data defines SVM
  - Performance on (unseen) test points
  - Each bit of IP an input feature

Results

Results

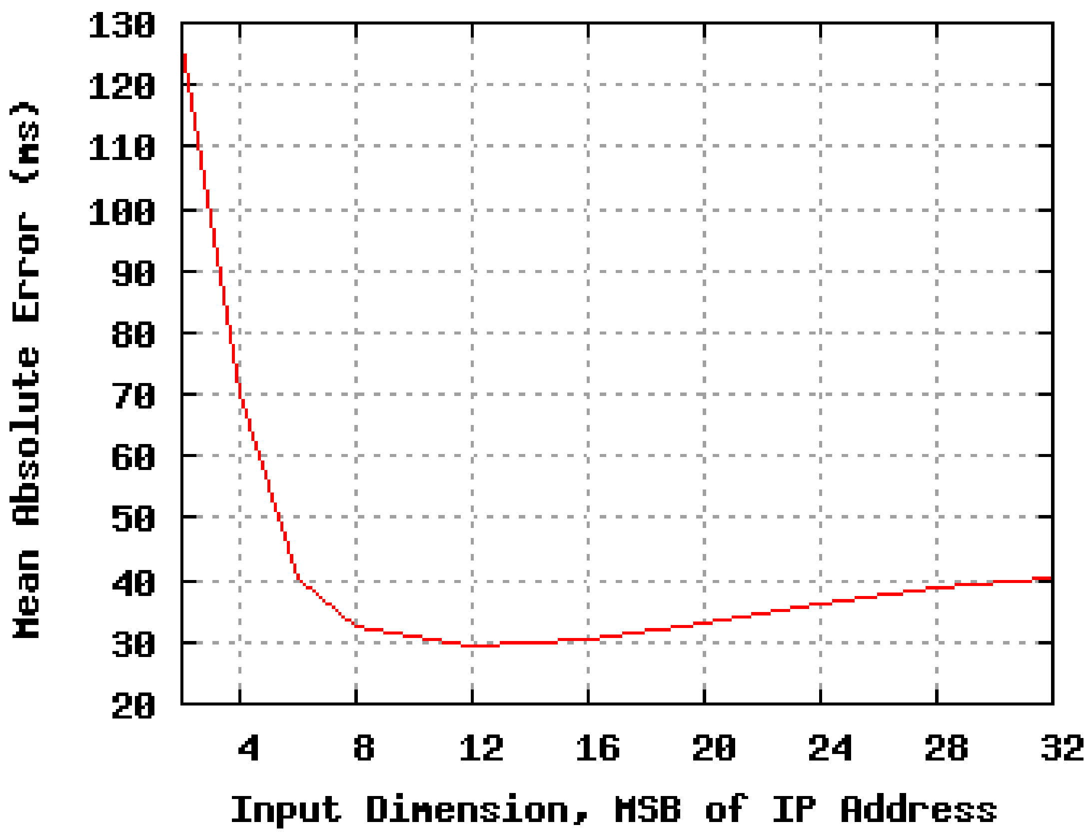
- Spoiler: So, does it work?
- Yes, within 30% for more than 75% of predictions
- Performance varies with selection of parameters (multi-optimization problem)
  - Training Size
  - Input Dimension
  - Kernel

Training Size



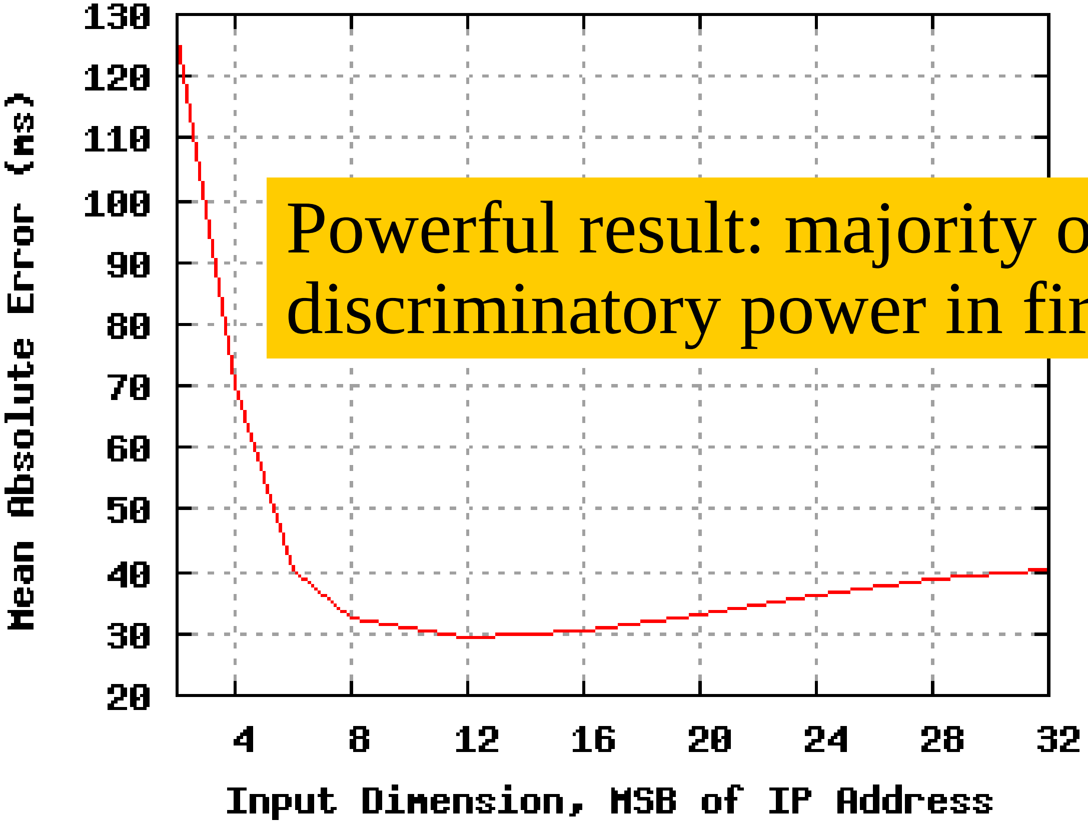
Question: Are MSB Better Predictors

- Determine error versus number most significant bits of test input IPs



Question: Are MSB Better Predictors

- Determine error versus number most significant bits of test input IPs



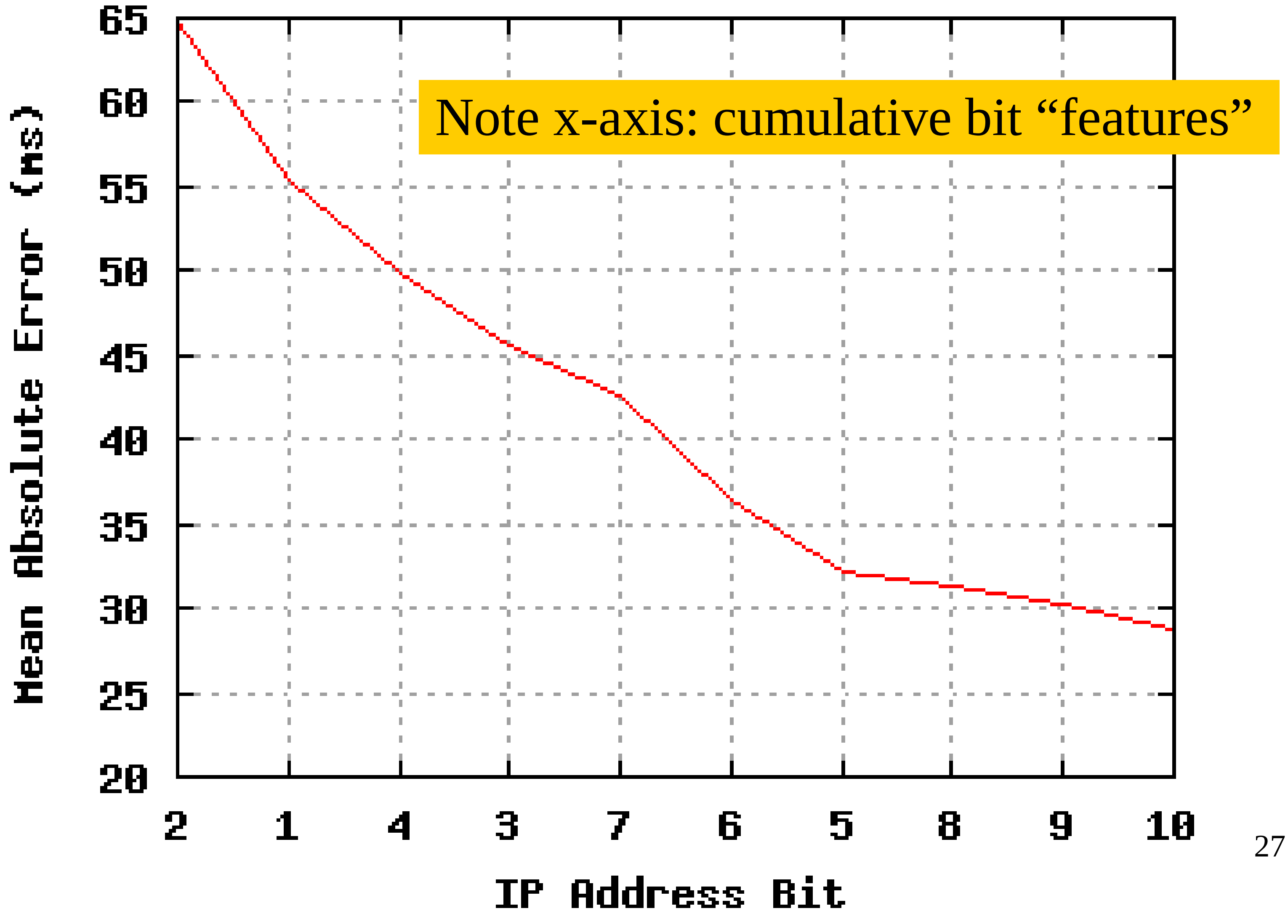
Powerful result: majority of discriminatory power in first 8 bits

Feature Selection

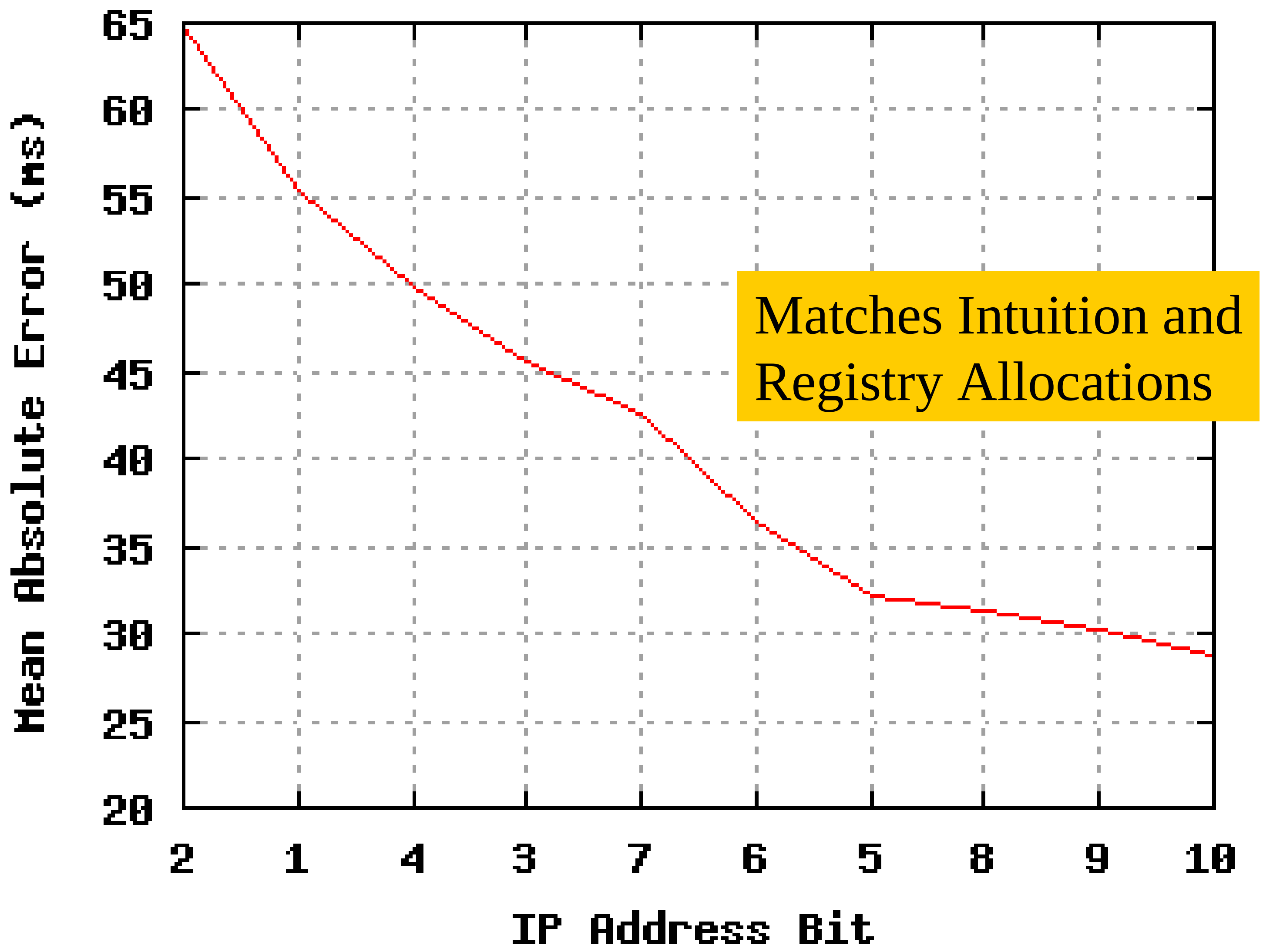
- Use feature selection to determine which individual bits of address contribute to discriminatory power of prediction

$$\theta_i \leftarrow \text{argmin}_j \ V(f(\boldsymbol{\theta},x_j),y) \ \ \forall \ x_j \ !\in \theta_1,\ldots,\theta_{i-1}$$

Feature Selection



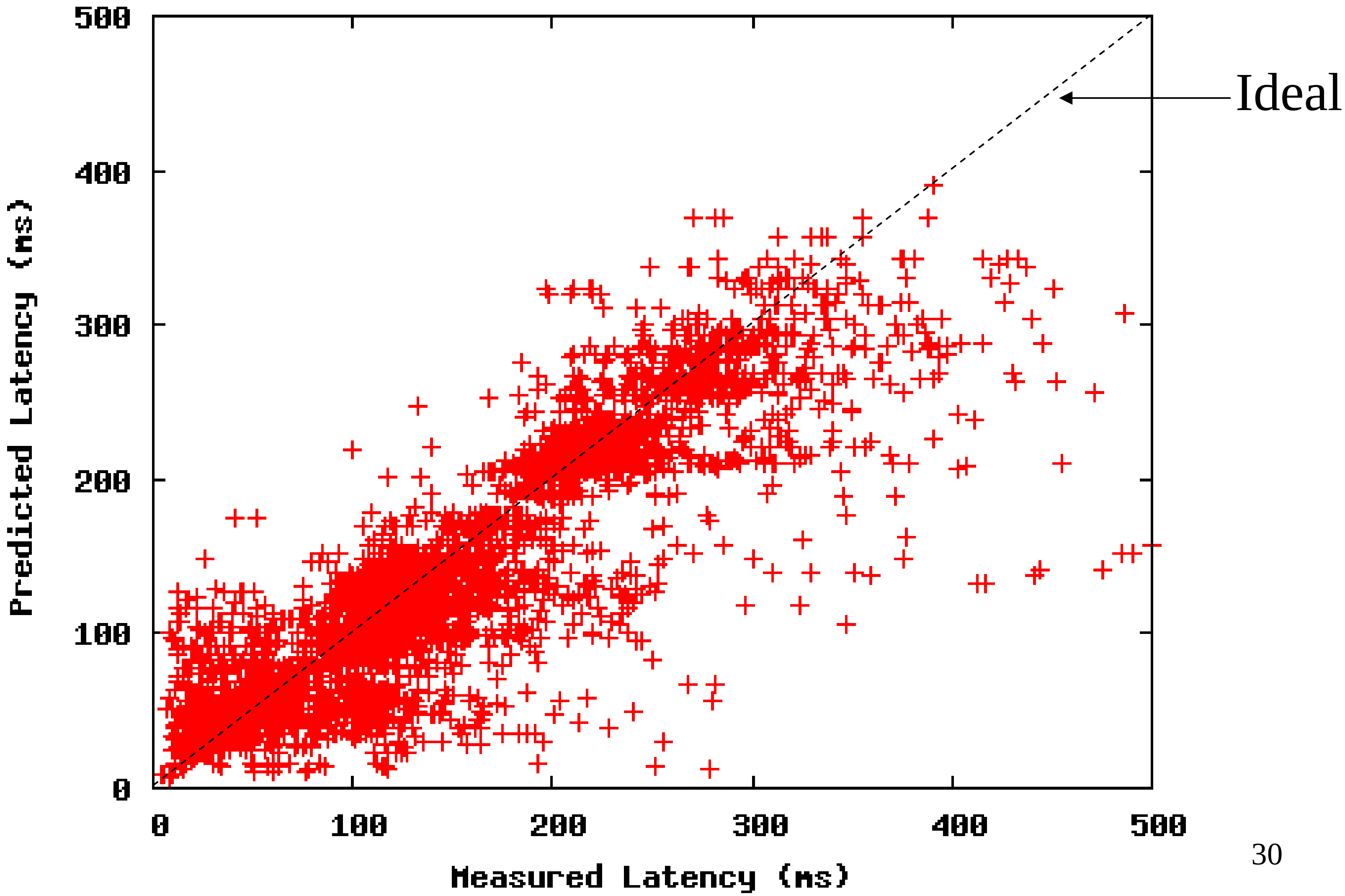
Feature Selection



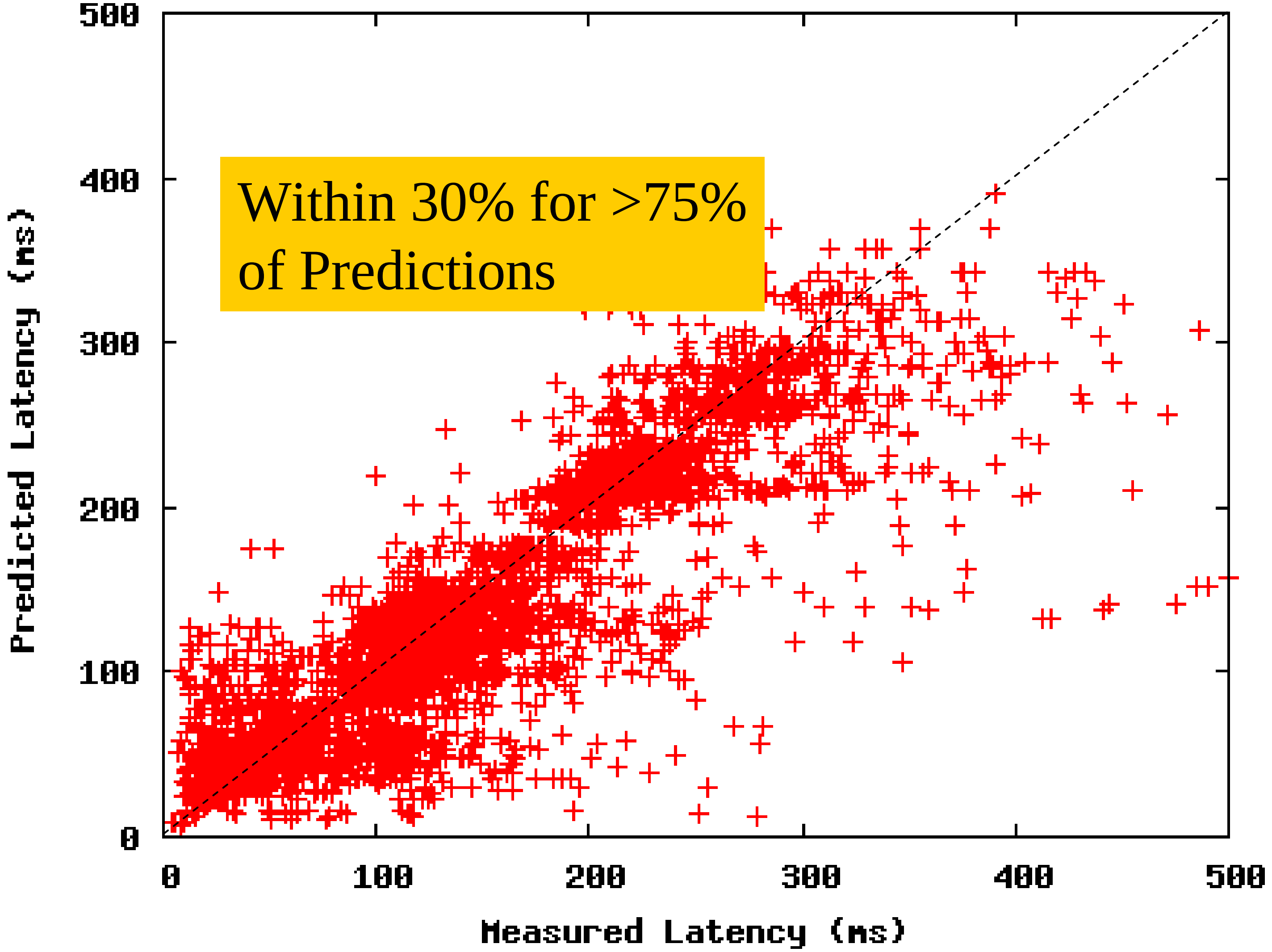
Performance

- Given empirically optimal training size and input features
- How well can agents predict latency to unknown destinations?

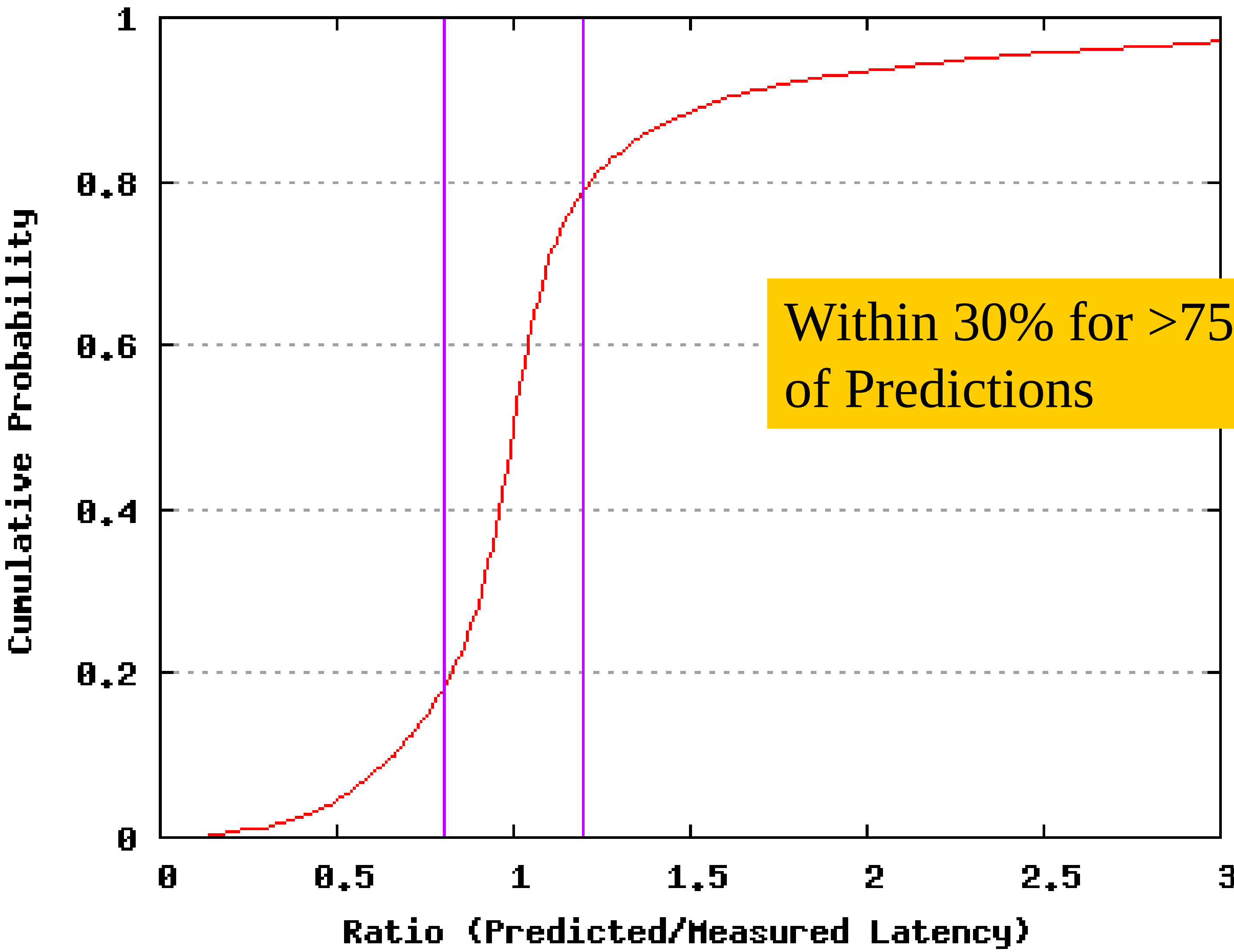
Prediction Performance



Prediction Performance



Prediction Performance



Within 30% for >75% of Predictions

Going Forward

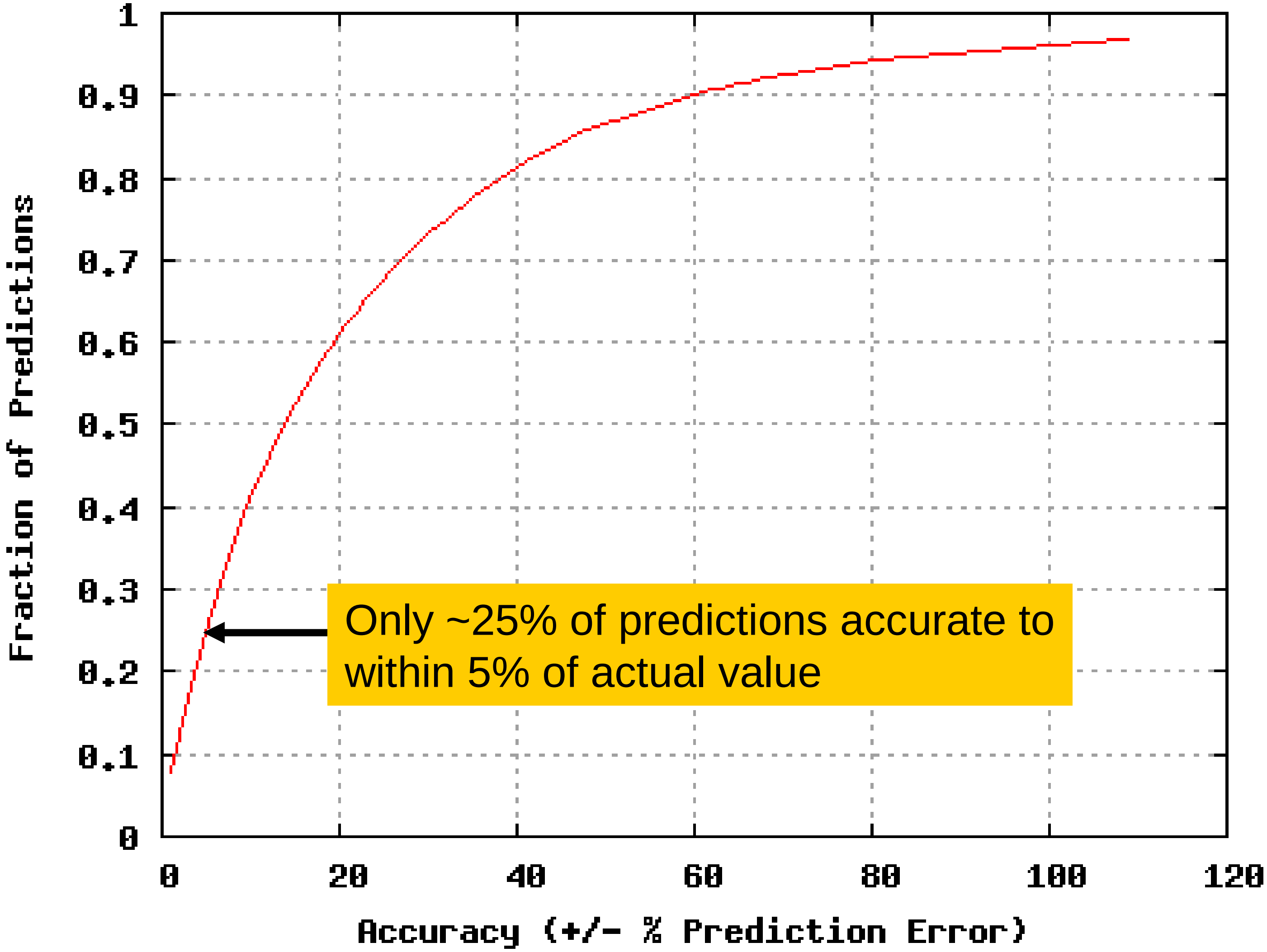
Future Research

- How agents select training data (random, BGP prefix, registry allocation, from TCP flows, etc)
- How performance decays over time and how often to retrain
- Online, continuous learning

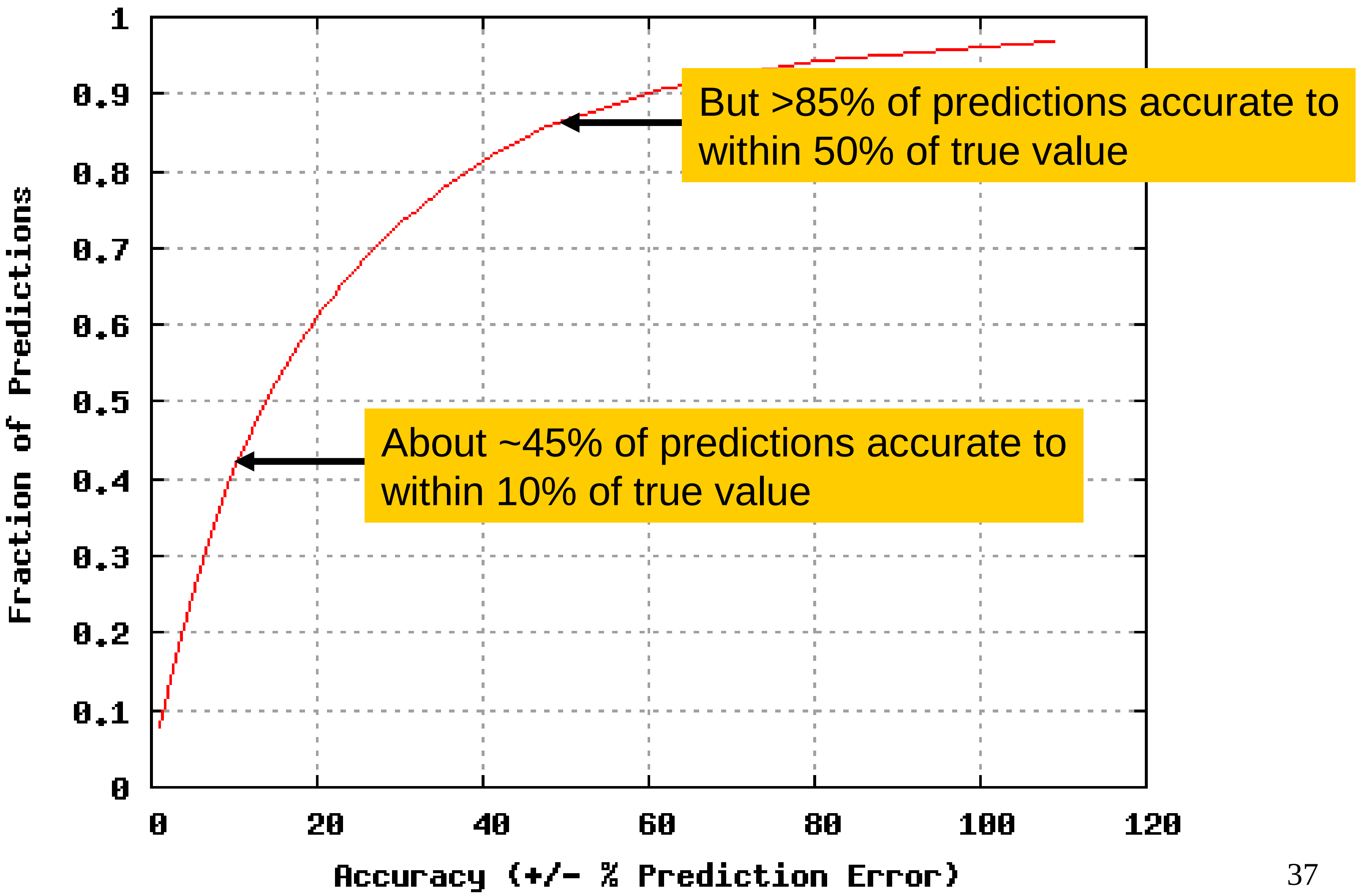
Summary - Questions?

- Major Results:
  - An agent-centric approach to latency prediction
  - Validation of SVMs and Kernel Functions as a means to learn on the basis of Internet Addresses
  - Feature Selection analysis of IP address informational content in predicting latency
  - Latency estimation accuracy within 30% of true value for > 75% of data points

Prediction Accuracy



Prediction Accuracy



Prediction Accuracy