



Continuous Machine Learning over Streaming Data

The Story Continues...

Roger Barga, GM Amazon Web Services

Nina Mishra and Sudipto Guha, Principal Scientist(s) AWS

Kapil Chhabra, Rubrik Inc



All data originates in real-time!

127.0.0.1 user-identifier frank [10/Oct/2000:13:55:36 -0700] "GET /apache_pb.gif HTTP/1.0" 200 2326

Common Log Entry



Smart Textiles



Beacons

<R,AMZN,T,G,R1>

NASDAQ OMX Record

"SeattlePublicWater/Kinesis/123/Realtime" – 412309129140

MQTT Record



Health Monitors




Smart Buildings

```
{
  "payerId": "Joe",
  "productCode": "AmazonS3",
  "clientProductCode": "AmazonS3",
  "usageType": "Bandwidth",
  "operation": "PUT",
  "value": "22490",
  "timestamp": "1216674828"
}
```

Metering Record

```
<165>1 2003-10-11T22:14:15.003Z
mymachine.example.com evntslog - ID47
[exampleSDID@32473 iut="3" eventSource="Application"
eventID="1011"][examplePriority@32473 class="high"]
Syslog Entry
```

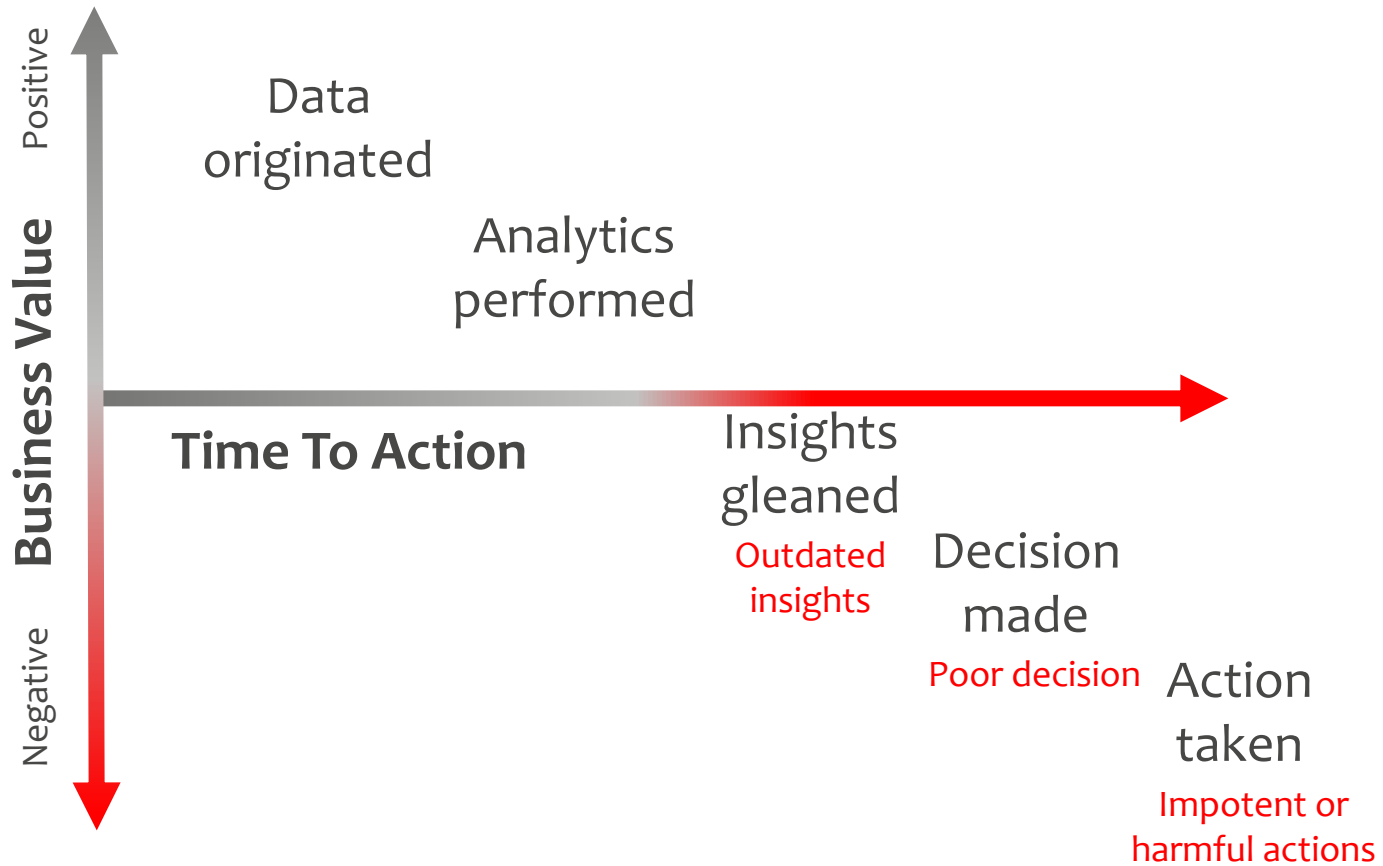


**But, analytics to gain insights is usually
done much, much later.**



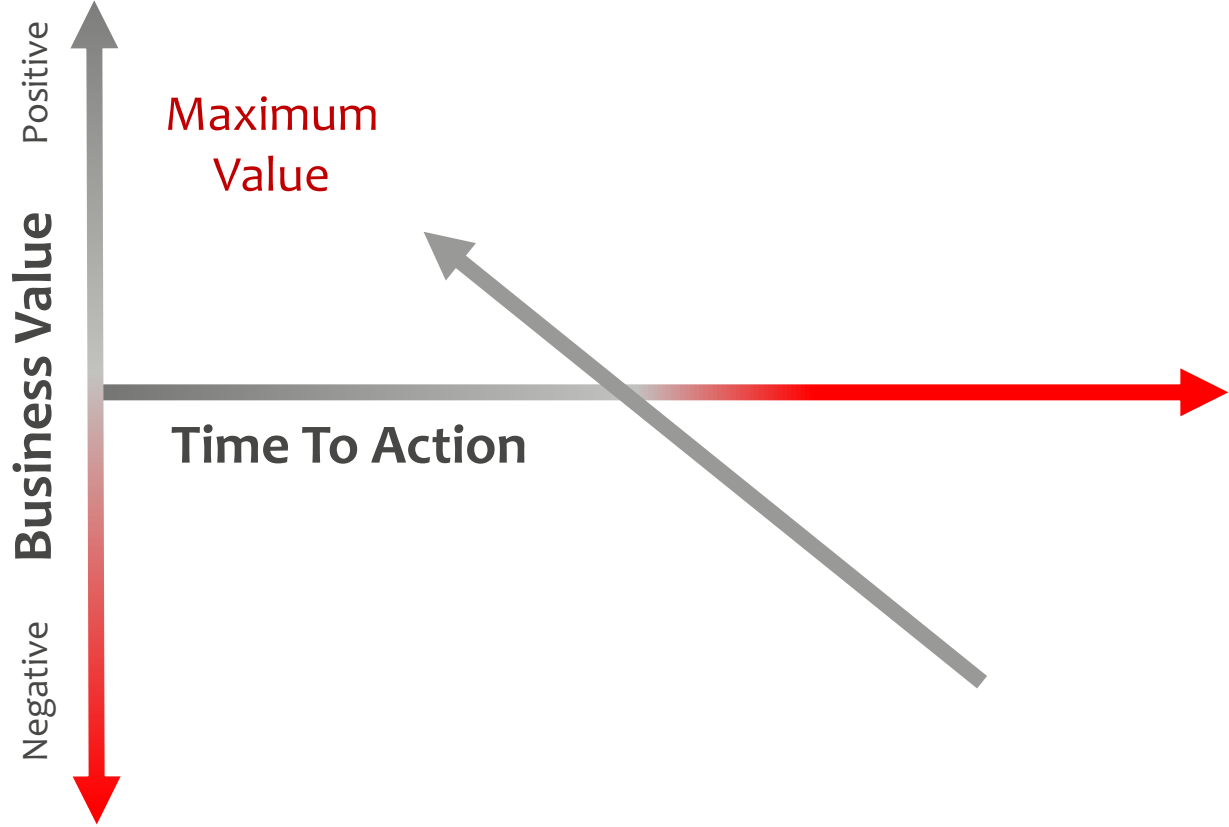
Insights are perishable.

Batch analytics operations take too long



Compress the analytics lifecycle

Maximize the value of data



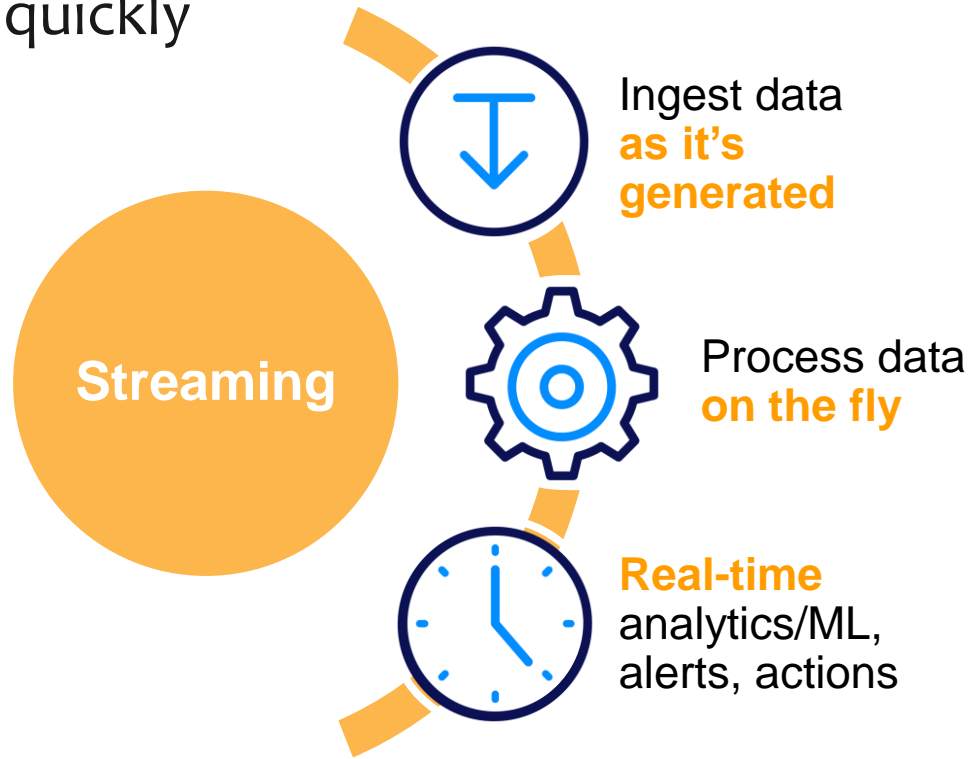
#Streaming



Streaming technology is necessary to detect and act on real-time perishable insights.

Kinesis Data Streaming Services

Get actionable insights quickly

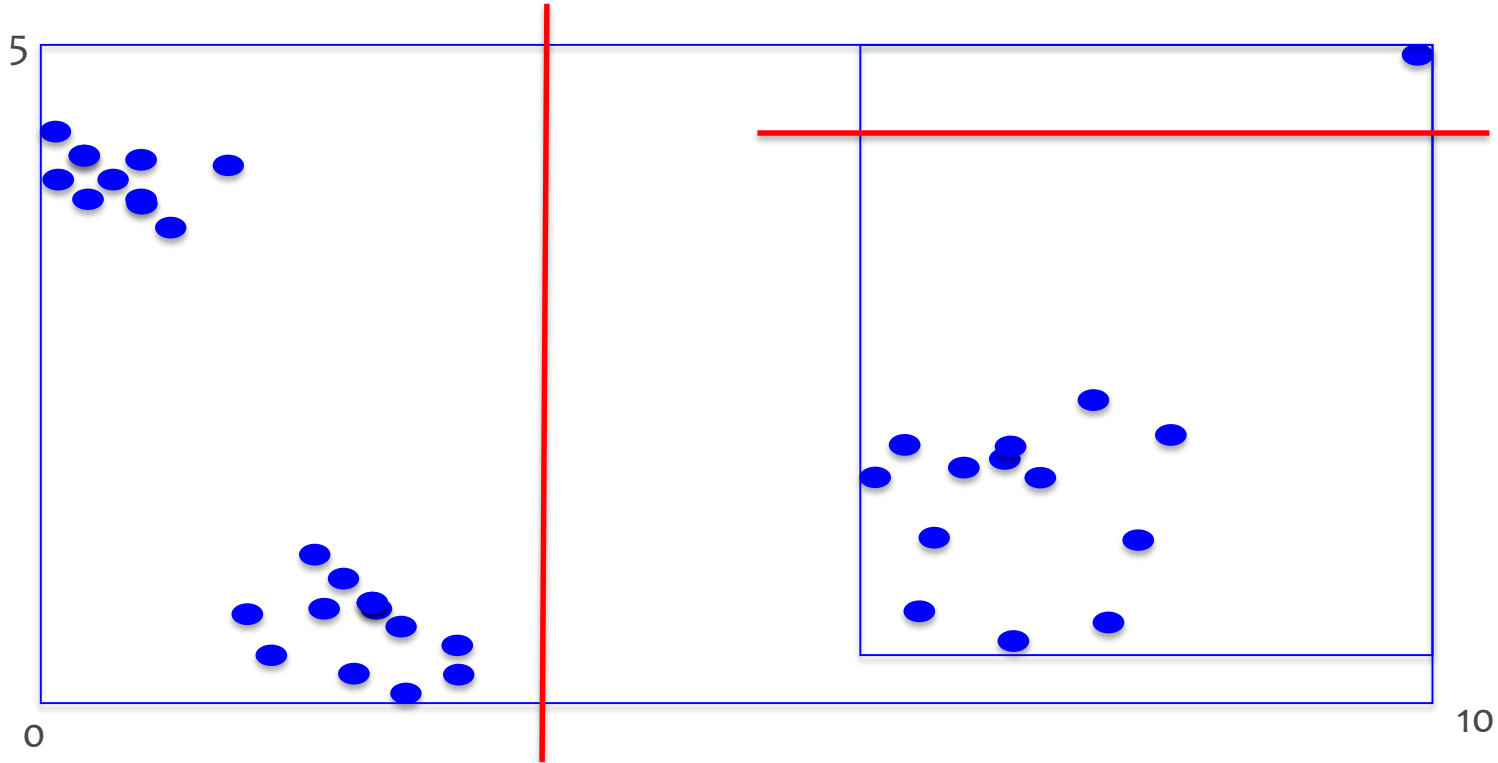




Robust Random Cut Forrest

Summary of a dynamic data stream, highly efficient, wide number of use cases...

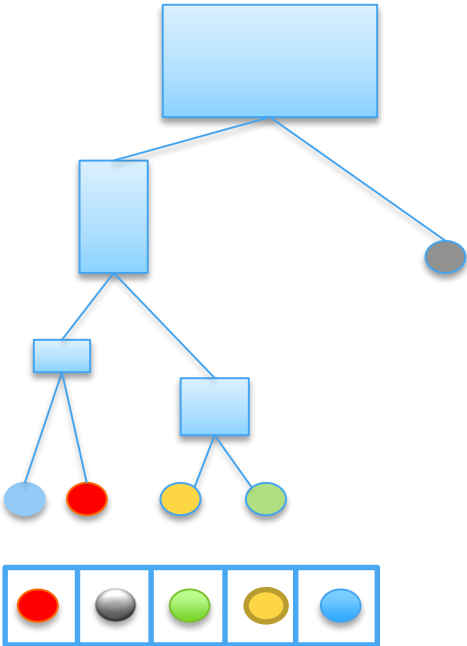
Random Cut Tree



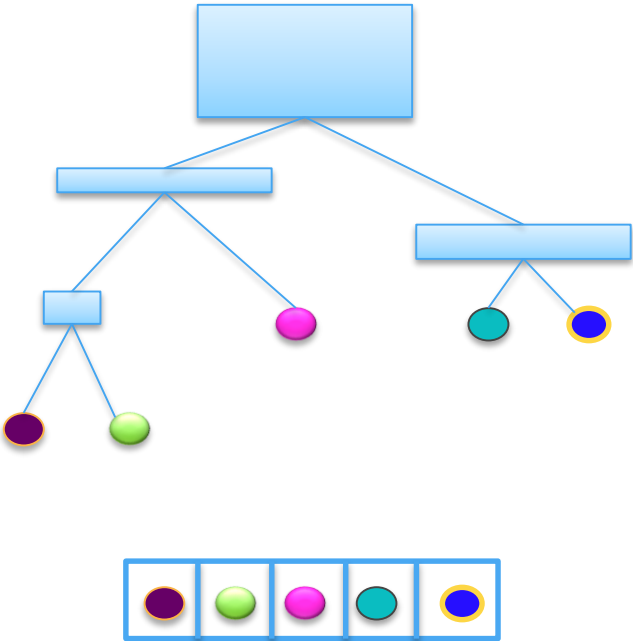
Range-biased Cut

Recurse: The cutting stops when each point is isolated.

Random Cut Forest



...

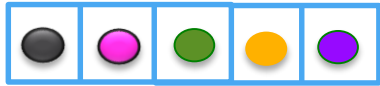


Each tree built on a random sample.

Random Sample of a Stream

Reservoir Sampling [Vitter]

Maintain random sample of 5 points in a stream?



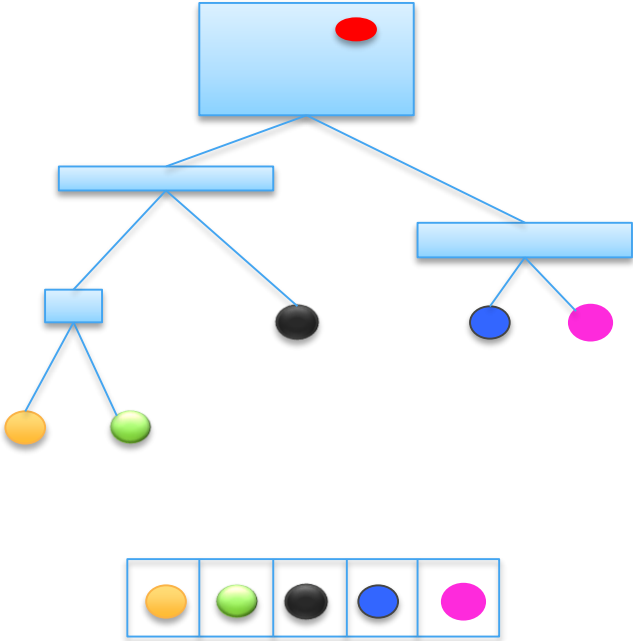
Keep 

heads with probability $\frac{5}{6}$

Discard 

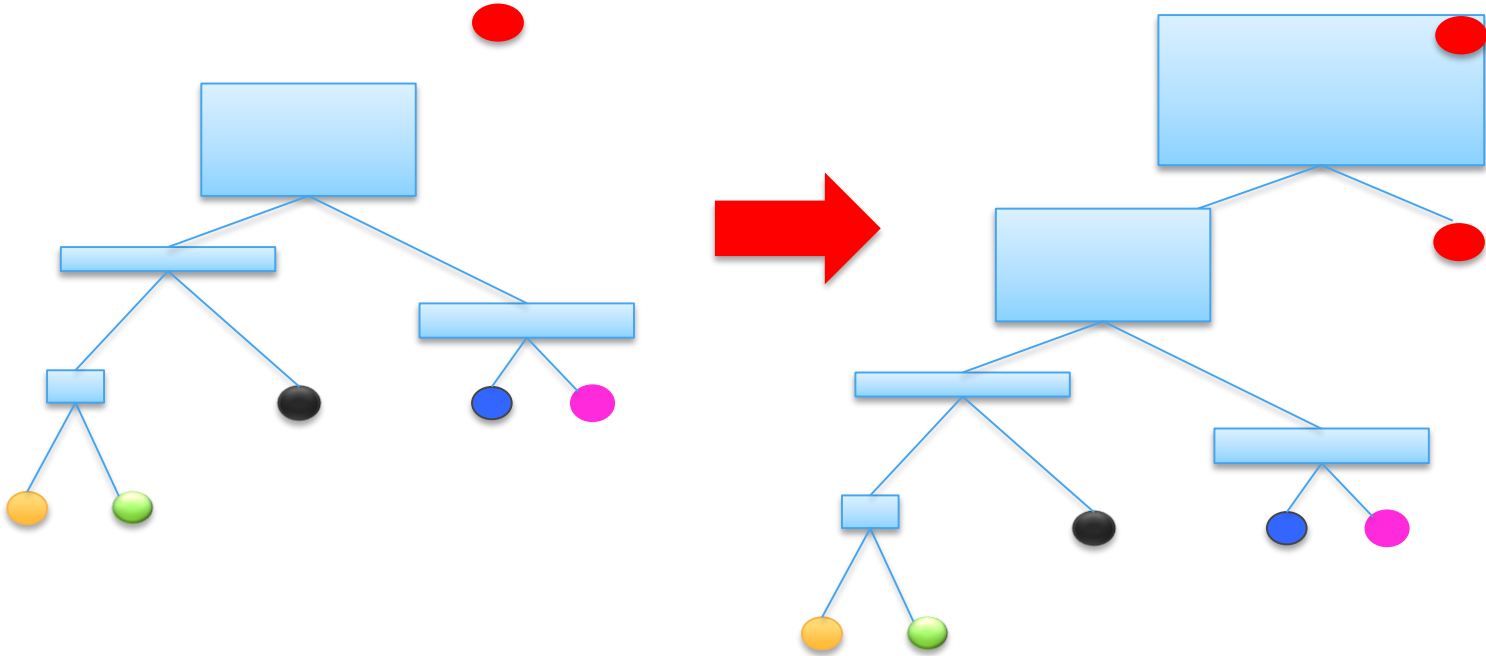
tails with probability $\frac{1}{6}$

Insert – Case I



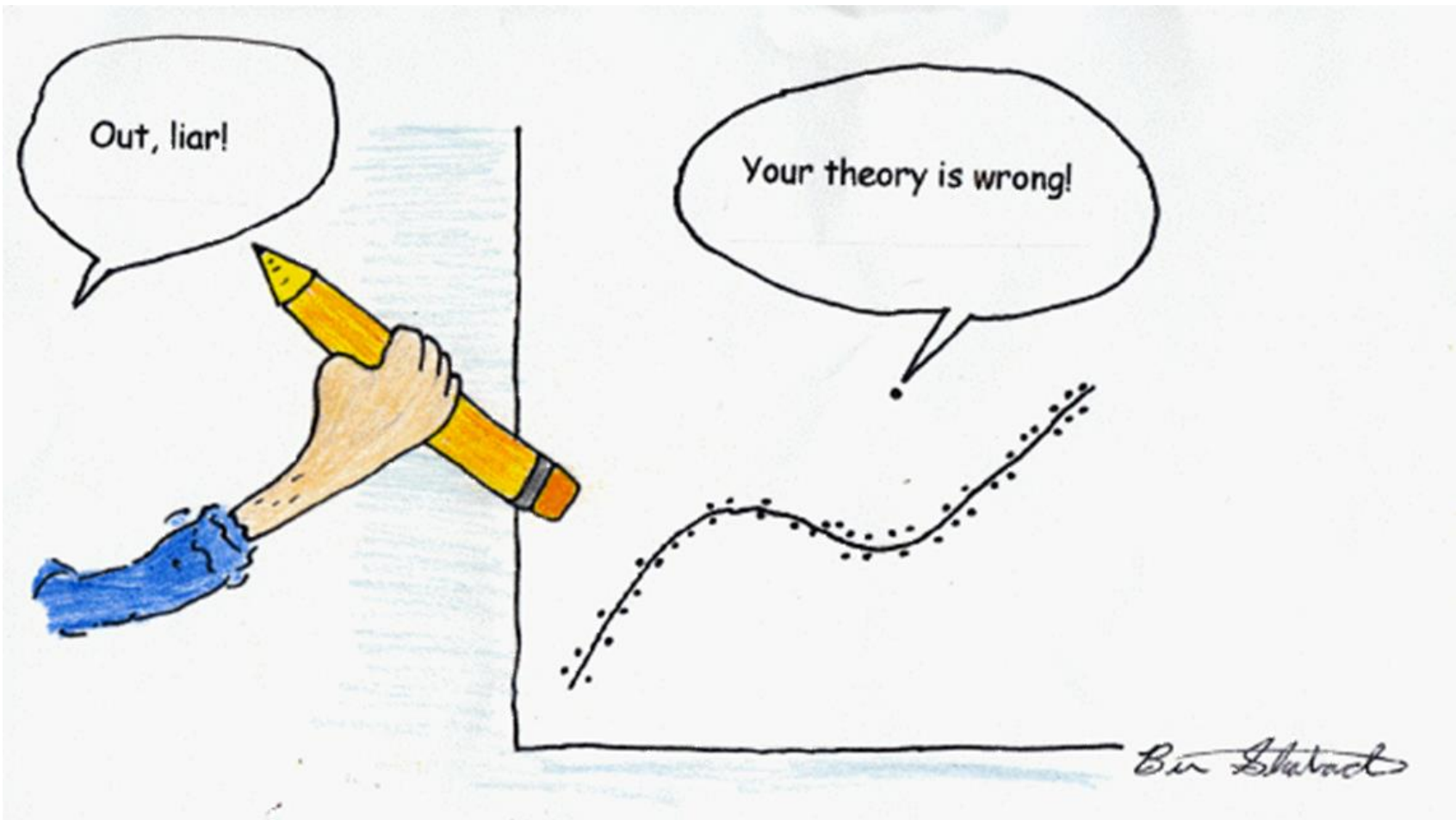
Start with the Root
If the point falls inside the bounding box
follow the path to the appropriate child

Insert – Case II



Theorem: Insert generates a tree $T' \sim T(\text{orange} \text{ green} \text{ black} \text{ blue} \text{ pink} \text{ red})$

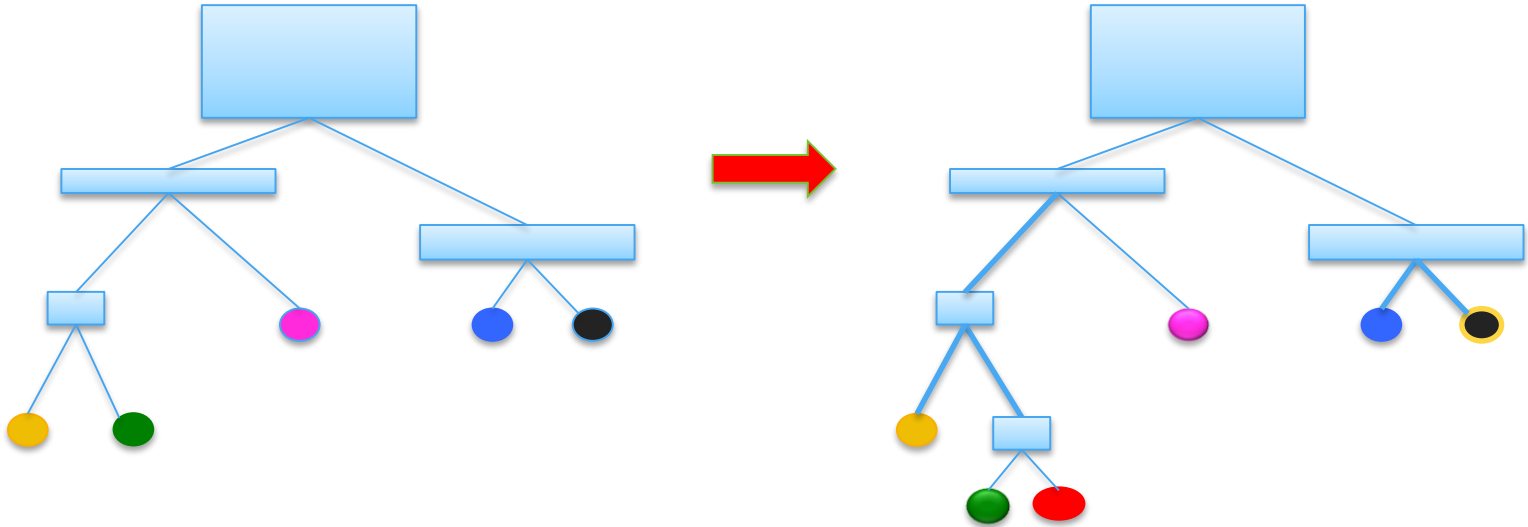
What is an Outlier?



Anomaly Score: Displacement

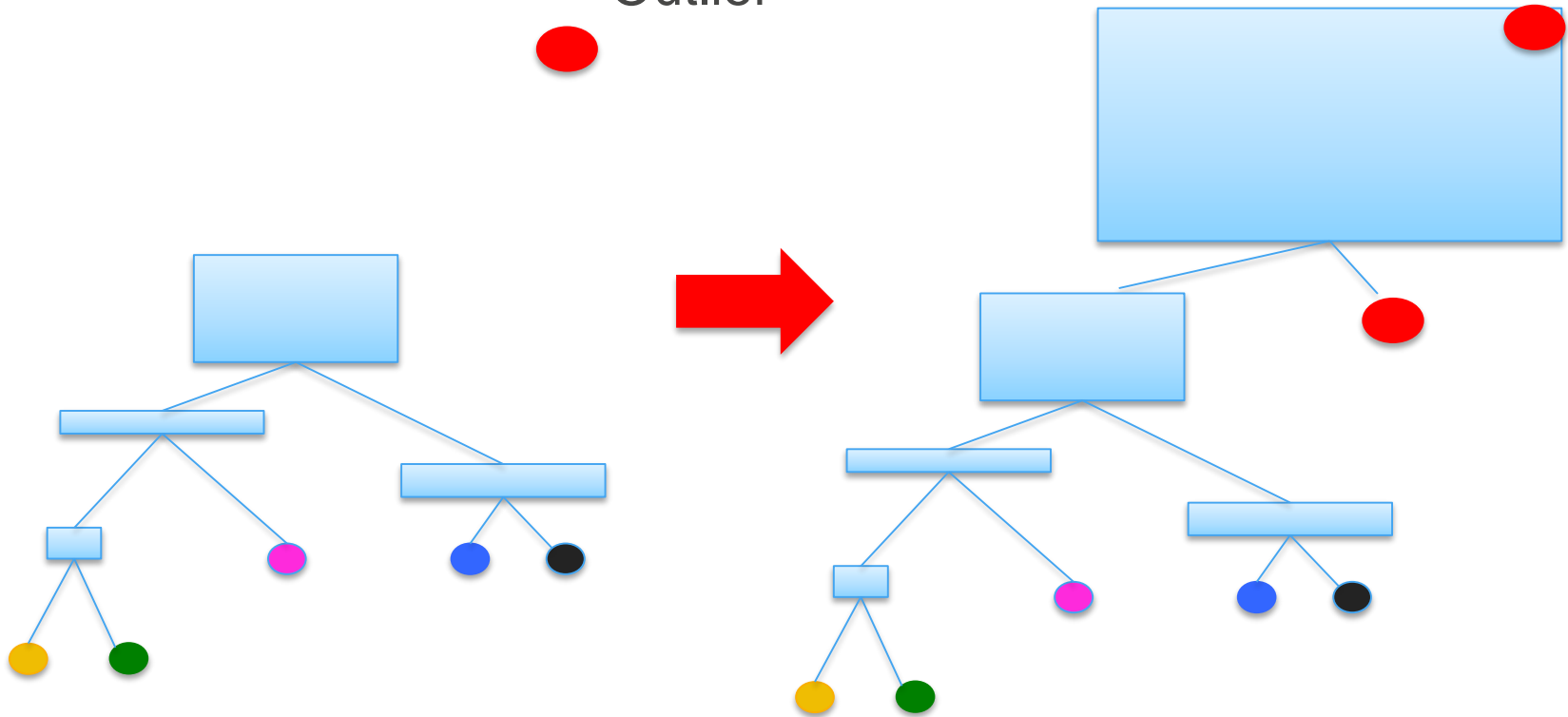
A point is an *anomaly* if its insertion greatly increases the tree size
(= sum of path lengths from root to leaves
= description length).

Inlier:



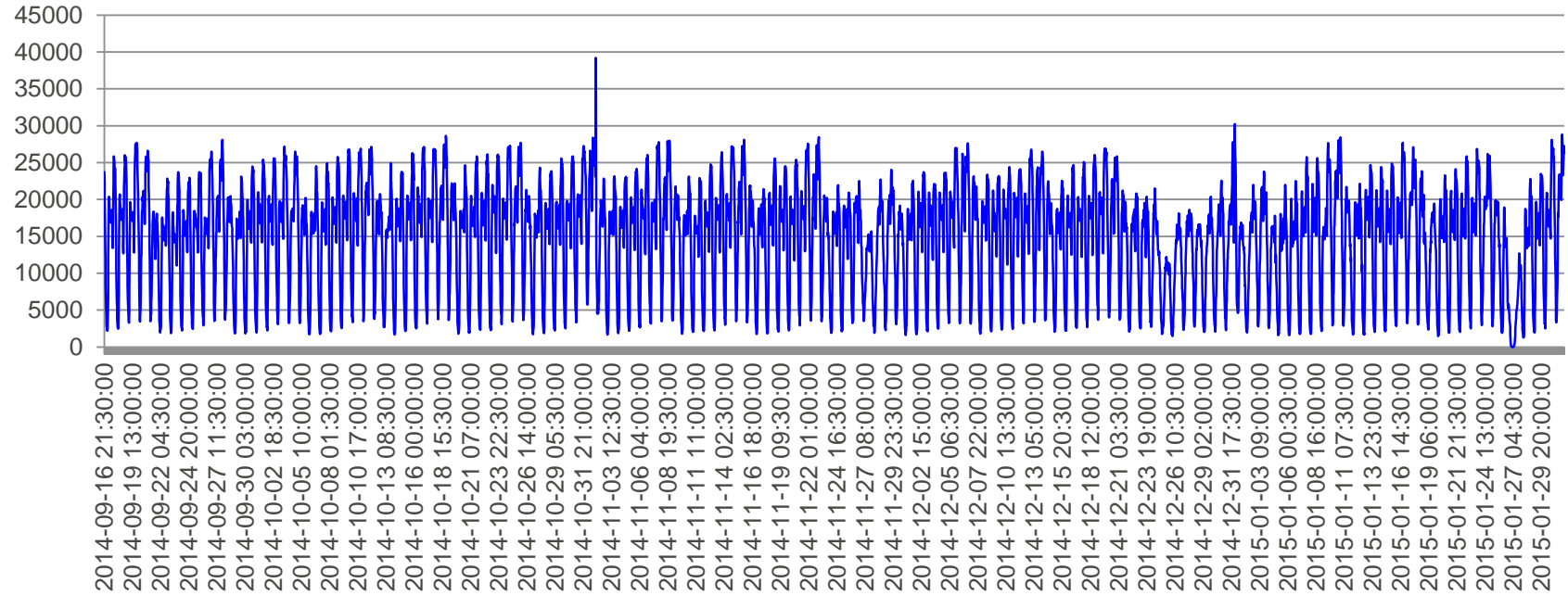
Anomaly Score: Displacement

Outlier



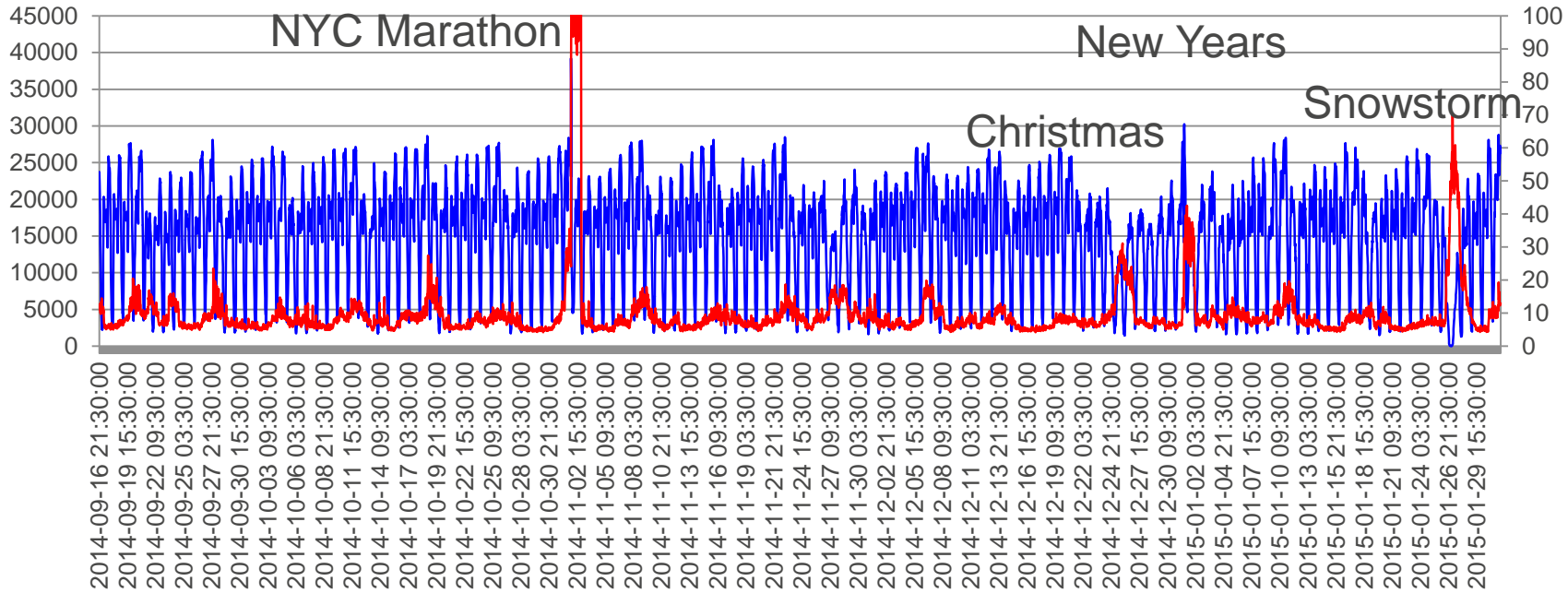
NYC Taxi Data

— numPassengers



NYC Taxi Data

— numPassengers — Anomaly Score



Robust Random Cut Forests

Quick Summary

- Random forests (RF) define ensemble models;
- The cut in RCF corresponds to specific choice of partitioning;
- Take a set of points, compute bounding box;
- Choose an axis proportional to the length (biased), then choose an uniform random cut in range;
- Recurse on both sides.
- Isolation Forests: Partition at random, dimensions unbiased;
- Why RCFs? Can maintain RCF tree distributions efficiently and do so online as data is streaming in. Distance preserving.

Robust Random Cut Forest Based Anomaly Detection On Streams

Sudipto Guha

University of Pennsylvania, Philadelphia, PA 19104.

SUDIPTO@CIS.UPENN.EDU

Nina Mishra

Amazon, Palo Alto, CA 94303.

NMISHRA@AMAZON.COM

Gourav Roy

Amazon, Bangalore, India 560055.

GOURAVR@AMAZON.COM

Okke Schrijvers

Stanford University, Palo Alto, CA 94305.

OKKES@CS.STANFORD.EDU

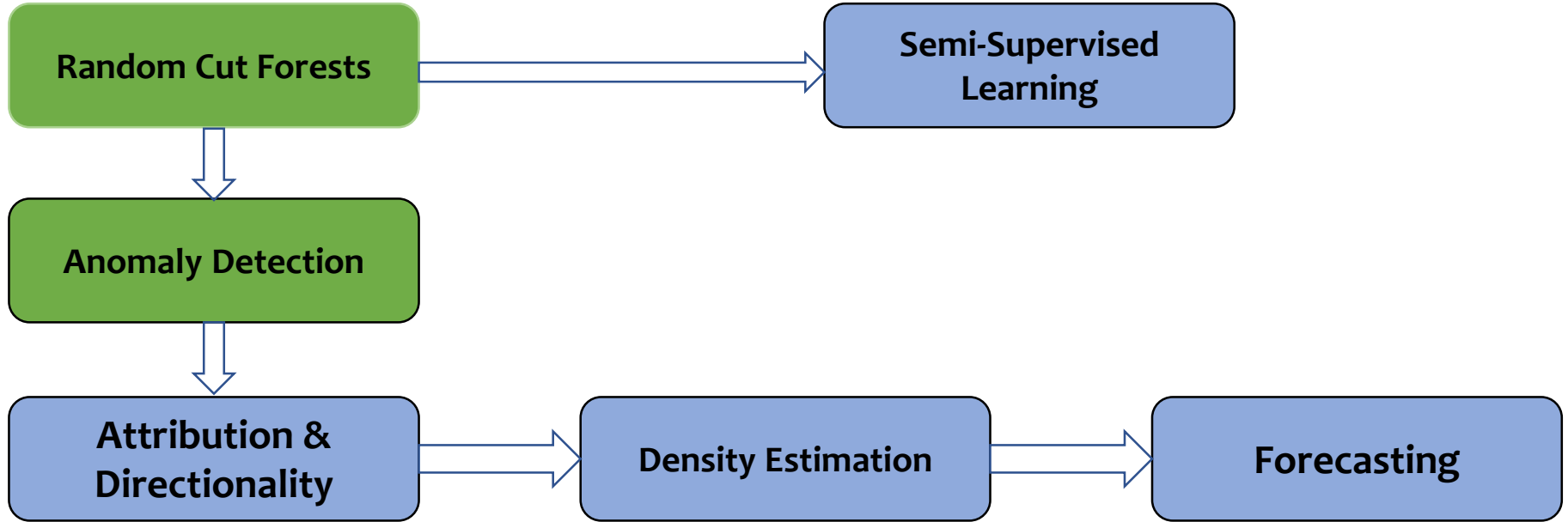
Abstract

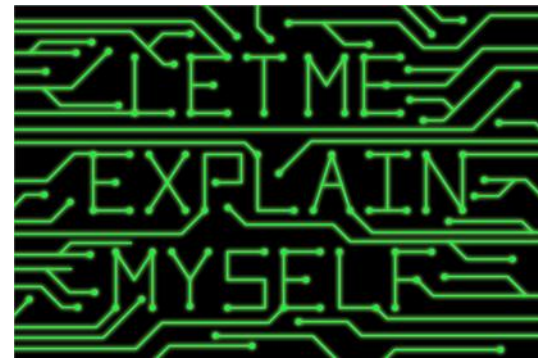
In this paper we focus on the anomaly detection problem for dynamic data streams through the lens of random cut forests. We investigate a robust random cut data structure that can be used as a sketch or synopsis of the input stream. We provide a plausible definition of non-parametric anomalies based on the influence of an unseen point on the remainder of the data, i.e., the externality imposed by that point. We show how the sketch can be efficiently updated in a dynamic data stream. We demonstrate the viability of the algorithm on publicly available real data.

a point is data dependent and corresponds to the externality imposed by the point in explaining the remainder of the data. We extend this notion of externality to handle “outlier masking” that often arises from duplicates and near duplicate records. Note that the notion of model complexity has to be amenable to efficient computation in dynamic data streams. This relates question (1) to question (2) which we discuss in greater detail next. However it is worth noting that anomaly detection is not well understood even in the simpler context of static batch processing and (2) remains relevant in the batch setting as well.

For question (2), we explore a randomized approach, akin to (Liu et al., 2012), due in part to the practical success reported in (Emmott et al., 2013). Randomization is a powerful tool and known to be valuable in supervised learn-

What we are going to see...





Explainable/Transparent/Interpretable ML

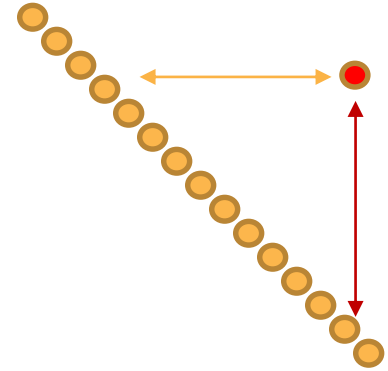
"If my time-series data with 30 features yields an unusually high anomaly score. How do I explain why this particular point in the time-series is unusual? [..] Ideally I'm looking for some way to visualize "feature importance" for a specific data point."

--- Robin Meehan, Inasight.com

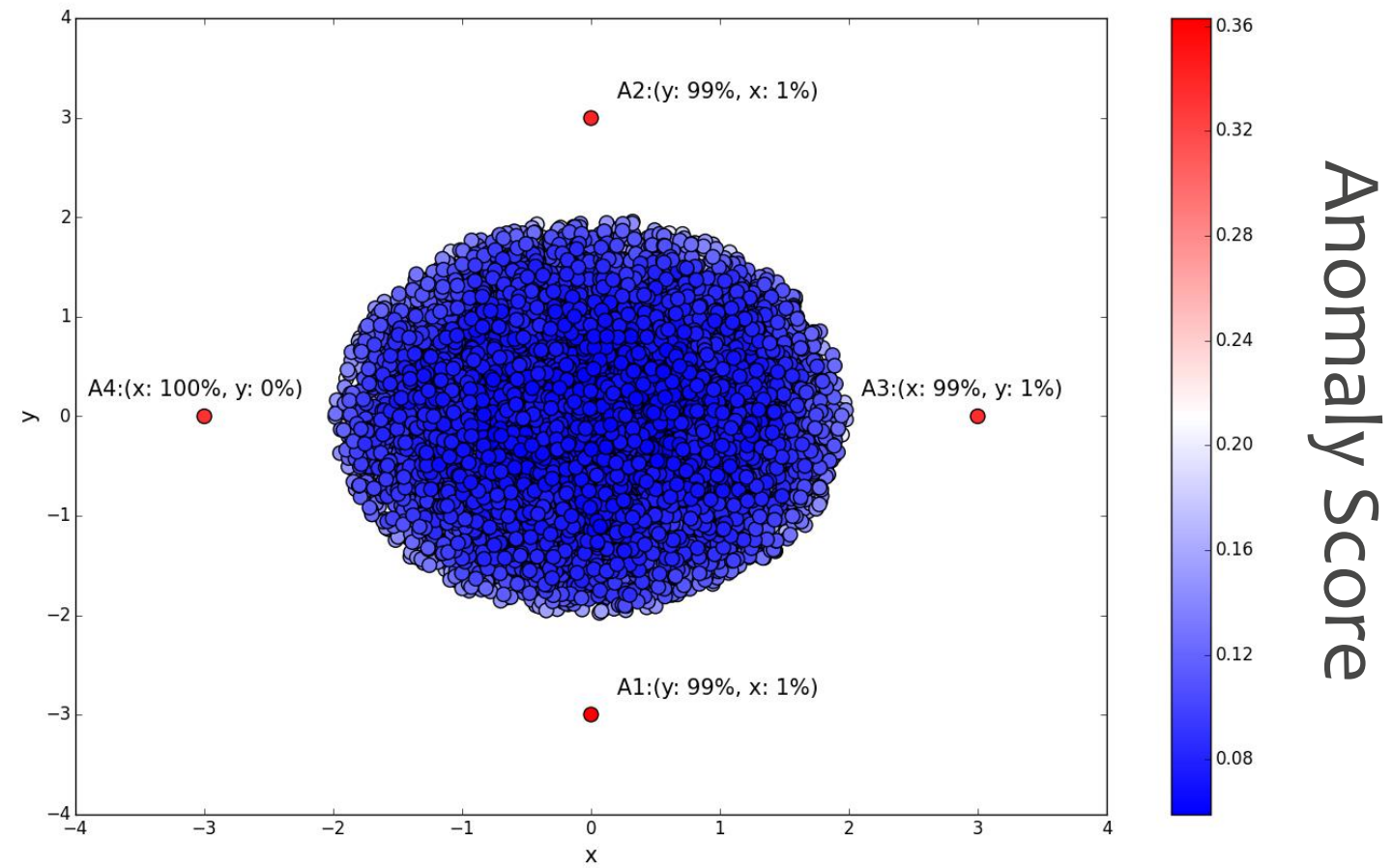
What is Attribution?

It's the ratio of the “distance” of the anomaly from normal.
(It's a distance in space of repeated patterns in the data.)

$$\Delta^i(p) = \frac{(\text{Score}^{+i}(p) - \text{Score}^{-i}(p))}{\text{Score}^{+i}(p)}$$



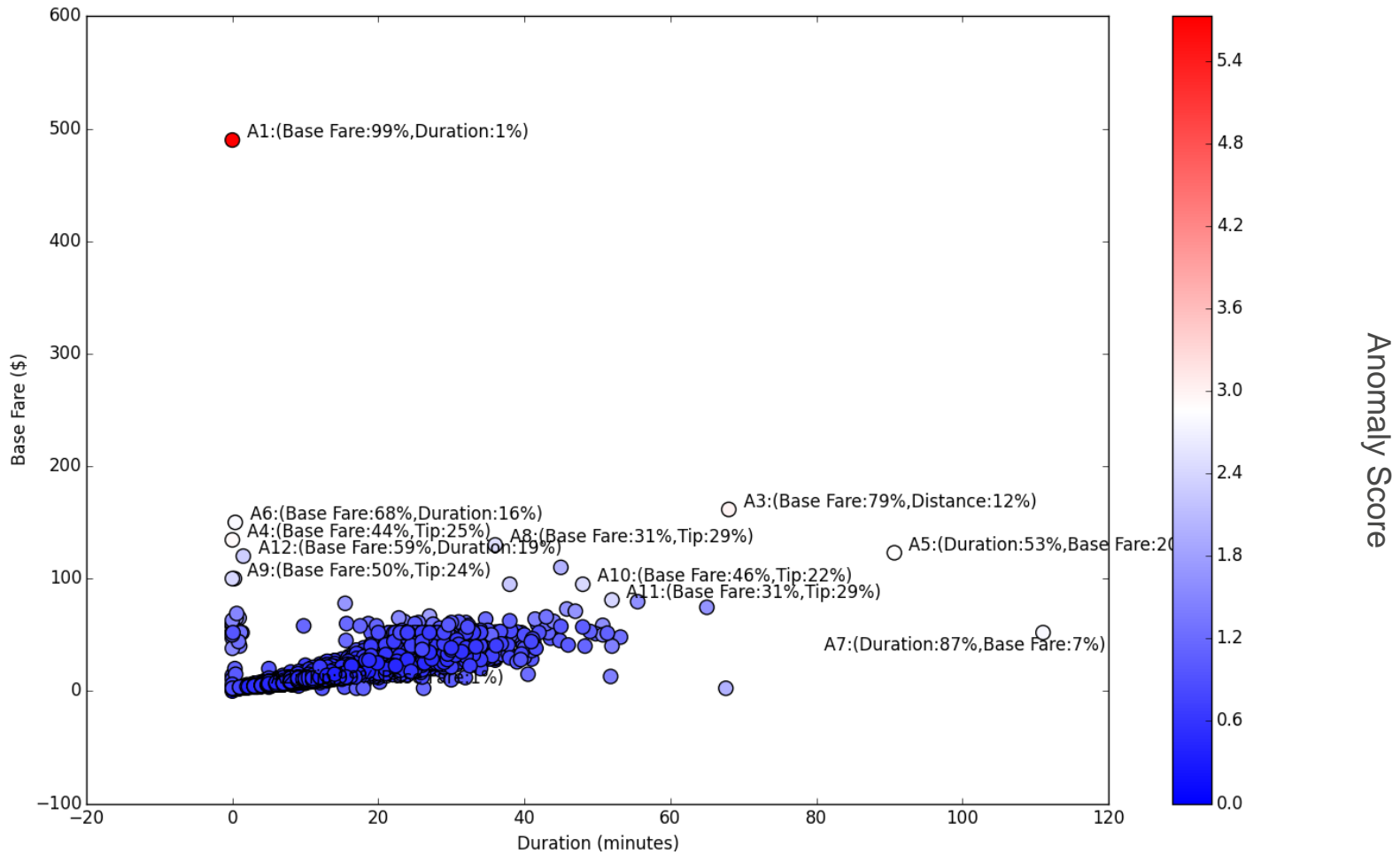
What is Attribution?



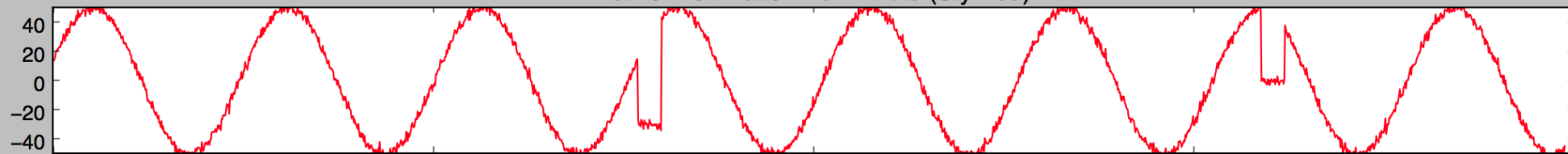
NYC Taxi Ridership Data¹

| Pickup Time | Dropoff Time | Distance | Base Fare | Surcharge | Tax | Tip | Tolls | Total |
|-------------|--------------|----------|-----------|-----------|-----|------|-------|-------|
| 7/1/14 1:43 | 7/1/14 1:51 | 2.59 | 9.5 | 0.5 | 0.5 | 0 | 0 | 10.5 |
| 7/1/14 1:33 | 7/1/14 1:47 | 2.38 | 12 | 0.5 | 0.5 | 0 | 0 | 13 |
| 7/1/14 1:37 | 7/1/14 1:50 | 2.87 | 11.5 | 0.5 | 0.5 | 0 | 0 | 12.5 |
| 7/1/14 1:35 | 7/1/14 1:50 | 4.68 | 16 | 0.5 | 0.5 | 4.95 | 0 | 21.95 |
| 7/1/14 1:25 | 7/1/14 1:49 | 6.72 | 23 | 0.5 | 0.5 | 0 | 0 | 24 |
| 7/1/14 1:30 | 7/1/14 1:50 | 5.04 | 18.5 | 0.5 | 0.5 | 0 | 0 | 19.5 |
| 7/1/14 0:17 | 7/1/14 0:24 | 2.53 | 9 | 0.5 | 0.5 | 0 | 0 | 10 |
| 7/1/14 0:06 | 7/1/14 0:22 | 2.48 | 12.5 | 0.5 | 0.5 | 2.6 | 0 | 16.1 |
| 7/1/14 0:17 | 7/1/14 0:24 | 1.81 | 7.5 | 0.5 | 0.5 | 0 | 0 | 8.5 |
| 7/1/14 1:38 | 7/1/14 1:50 | 6.26 | 19 | 0.5 | 0.5 | 1 | 0 | 21 |

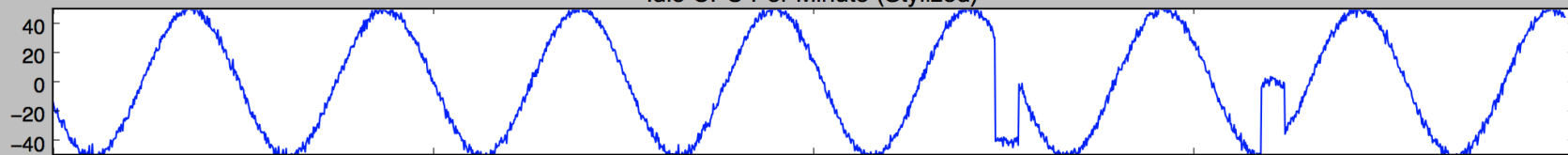
¹Public Data: http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml



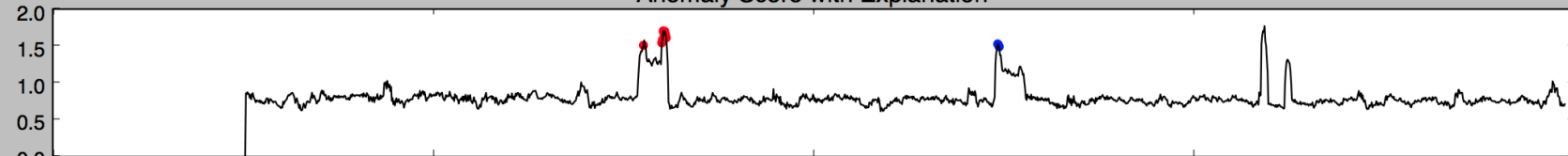
Network Utilization Per Minute (Stylized)



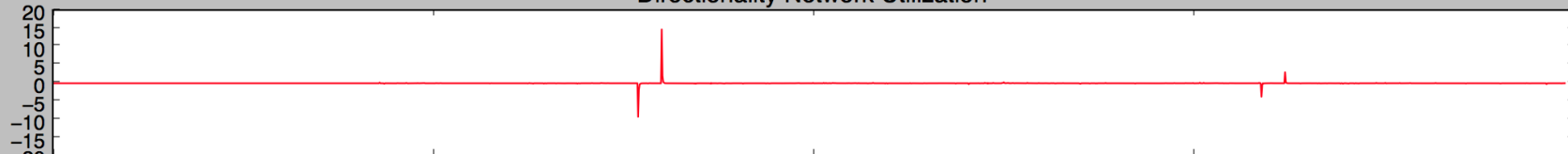
Idle CPU Per Minute (Stylized)



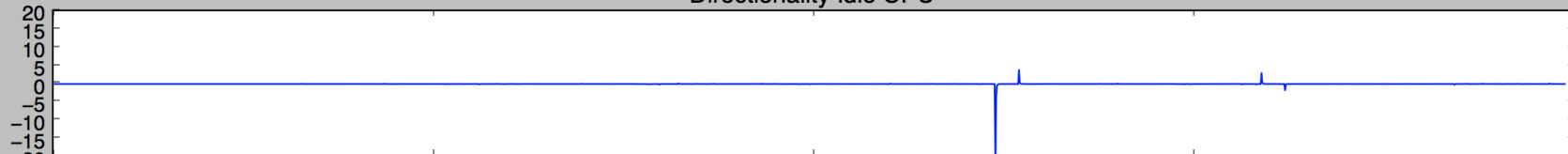
Anomaly Score with Explanation



Directionality Network Utilization



Directionality Idle CPU



0 500 1000 1500 2000

The Moving Example

A Fan/Turbine

1000 pts in each blade

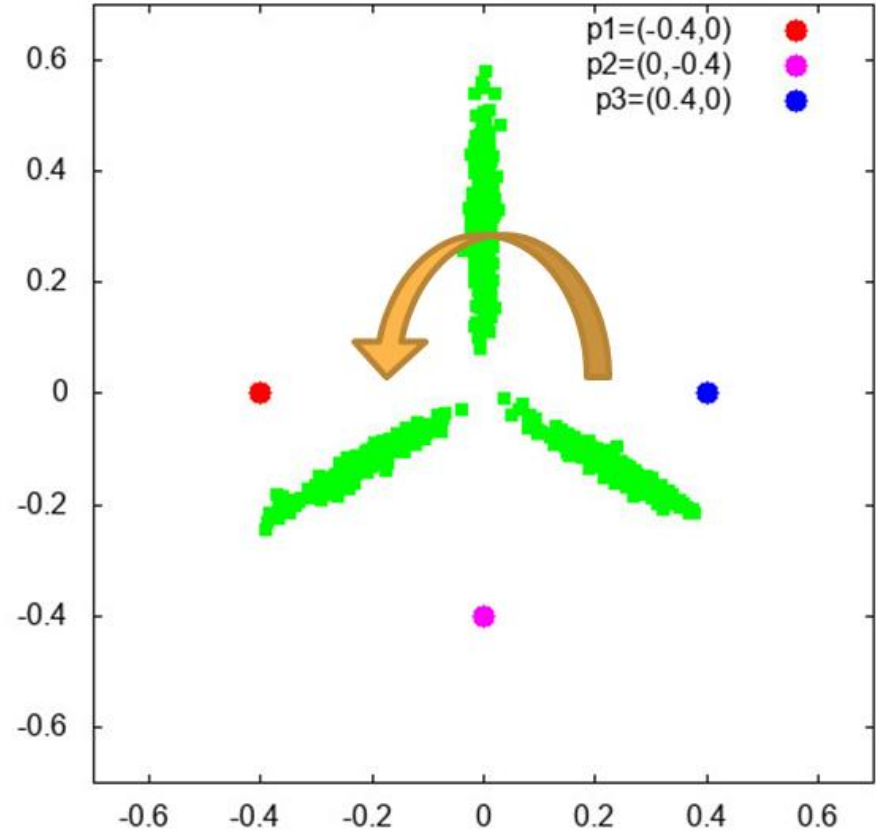
Gaussian, for simplicity

Blades designed unequal

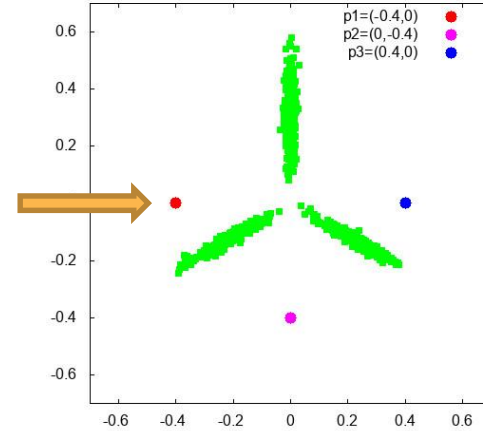
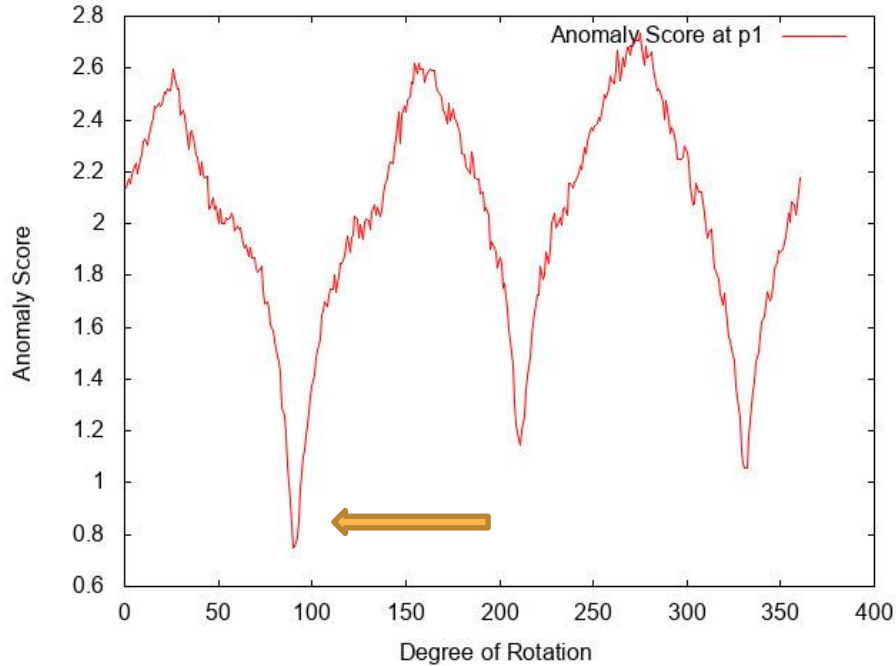
Rotate counterclockwise

3 special “query” points

100 trees, 256 points each



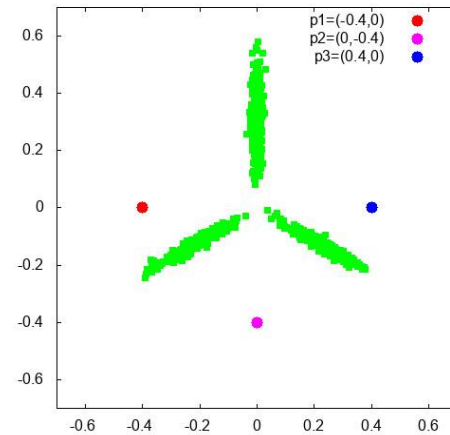
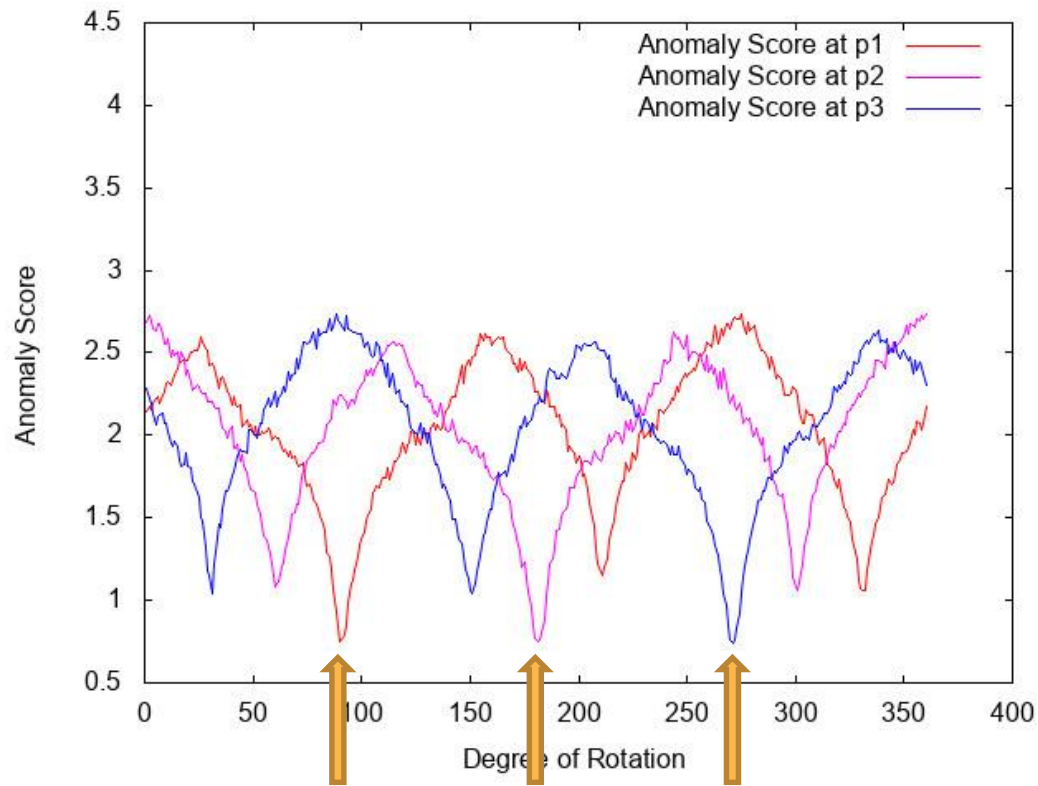
Anomaly Score at P1



Blade overhead = Not an anomaly

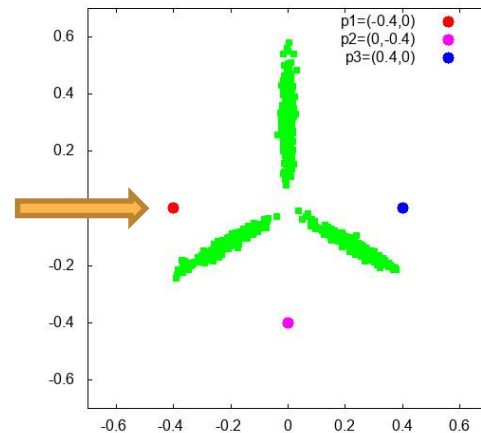
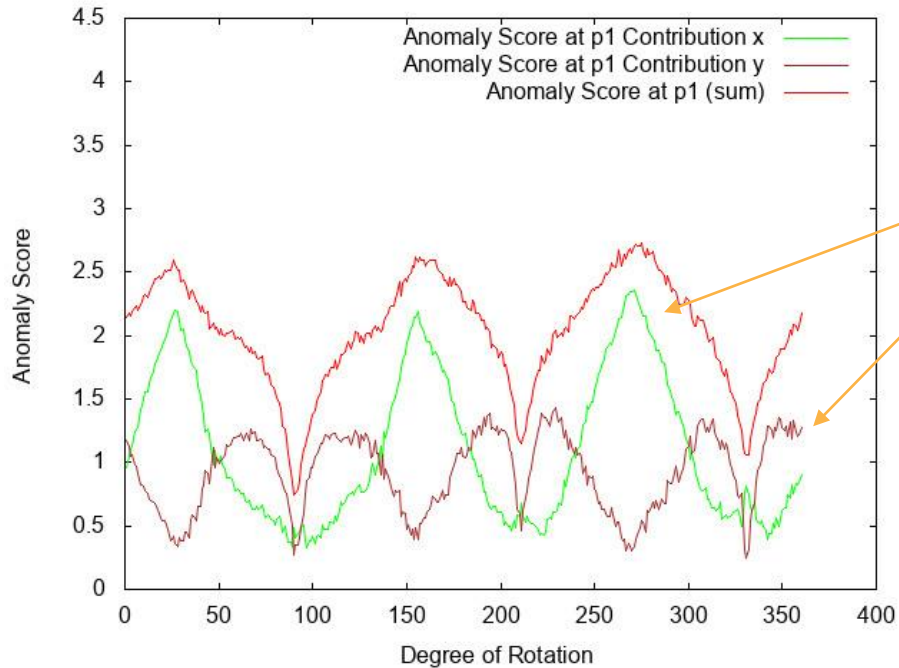
What is going on at 90 degrees?

All 3 Blades



Attribution

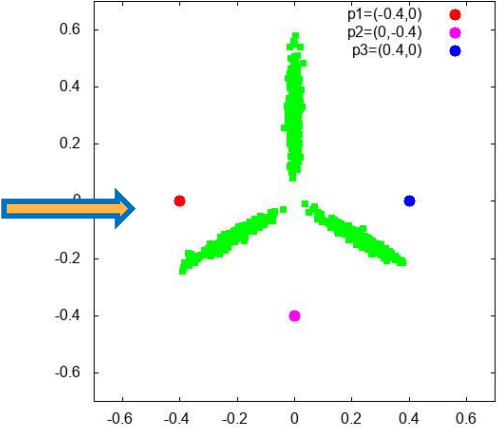
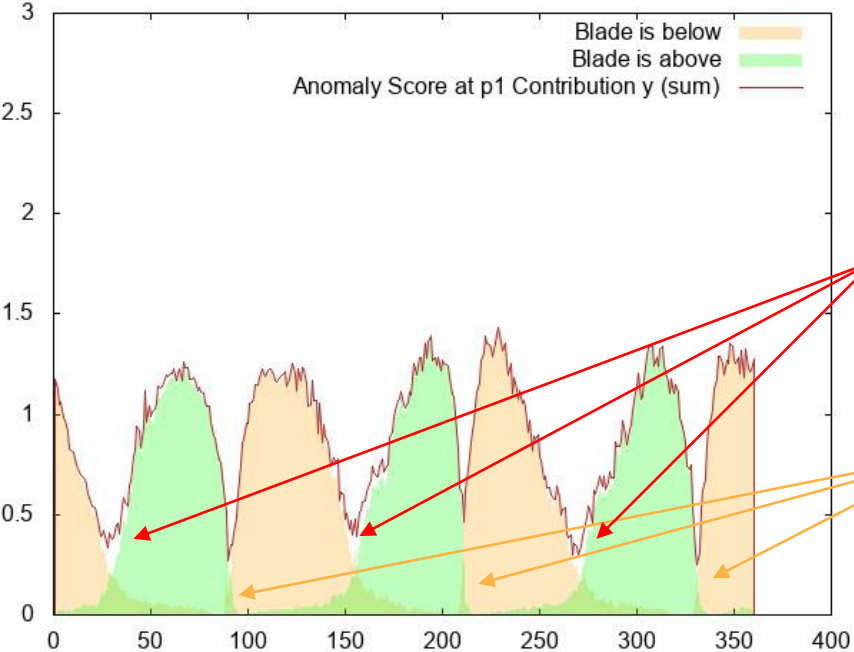
x coordinate's contribution for p1?



p1 is far away in x-coord most of the time

But what is happening to y?

Directionality

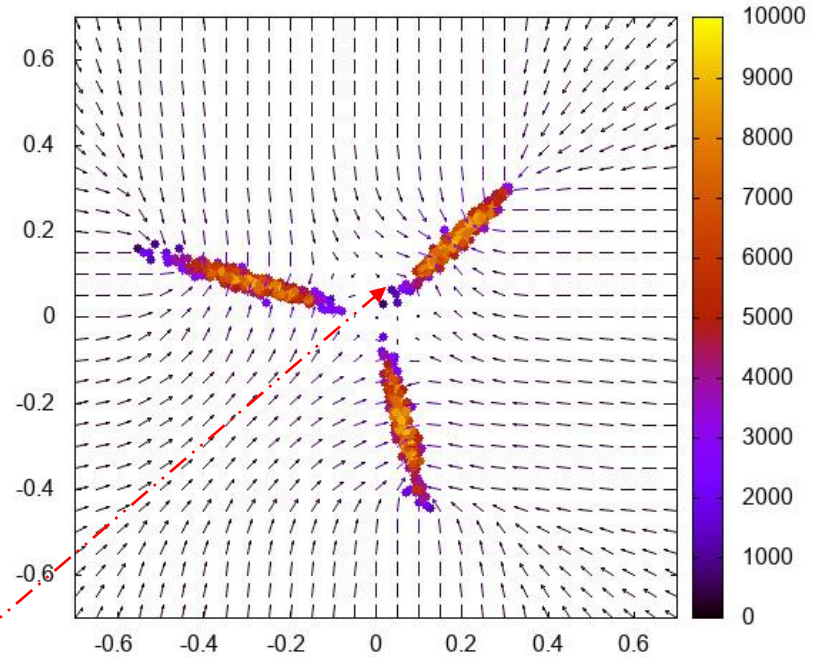
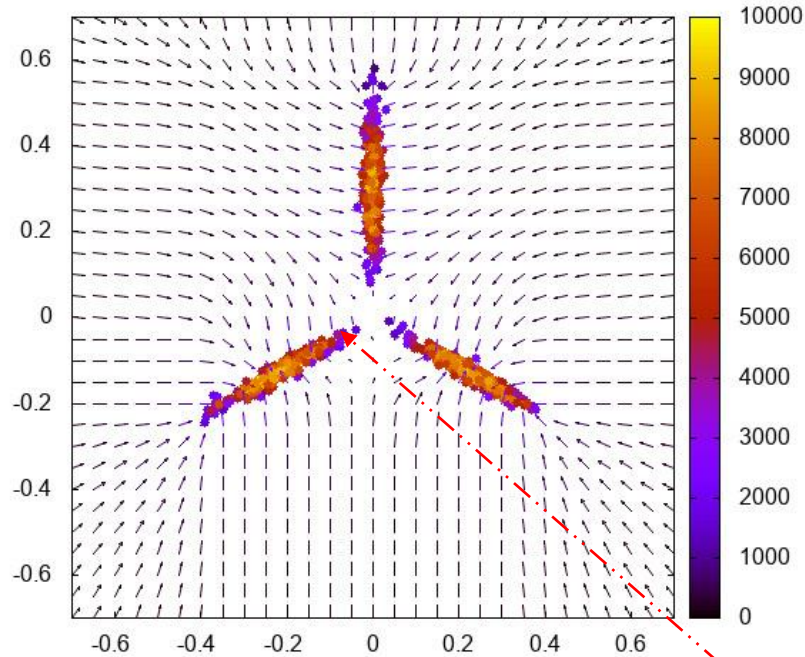


Slowly rotating away
Total score remains high

Sharp transition when the blade
moves from above to below at p1!
Total score plummets.

Initial Density and 75 Degrees Rotation

Directions towards higher density.



Bends towards center!

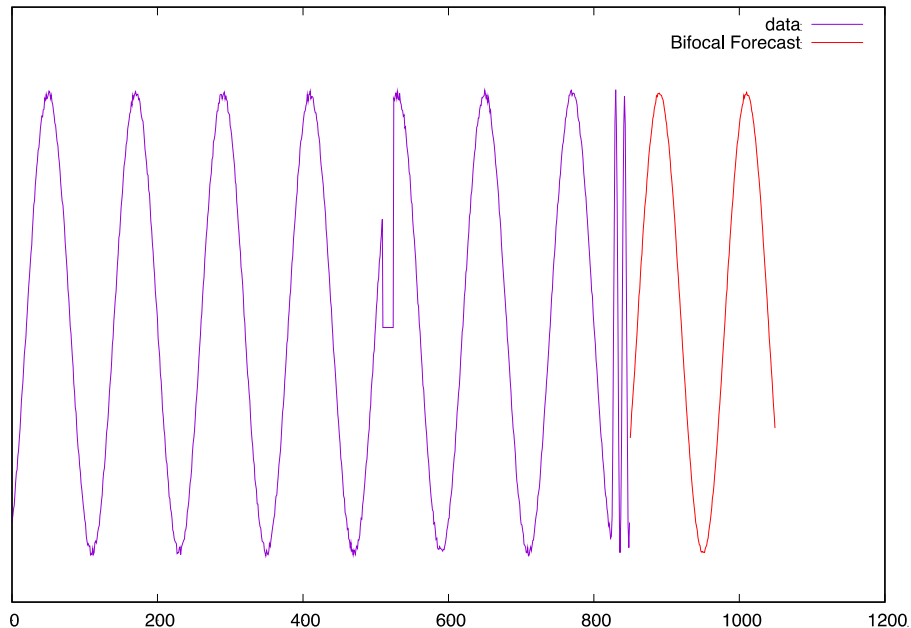
Forecasting

Exhibit B: Forecasting using RCFs

Not just the next value!

Ability to “see past” anomalies

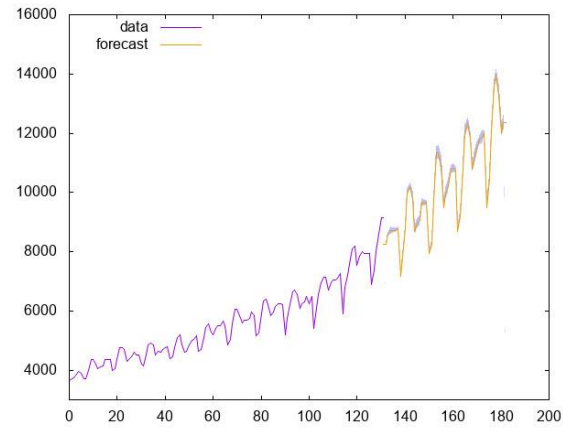
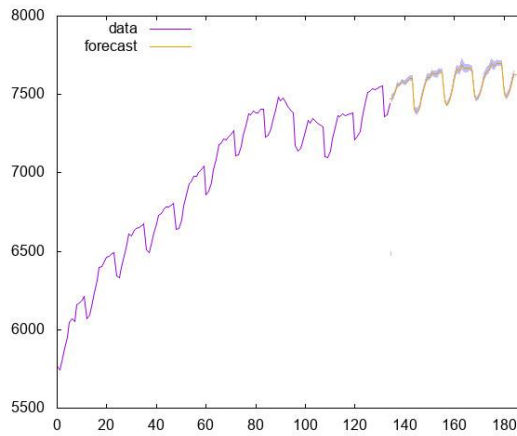
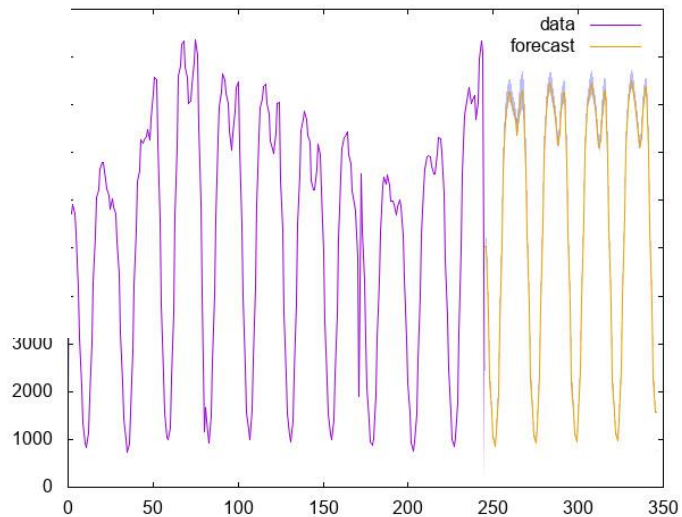
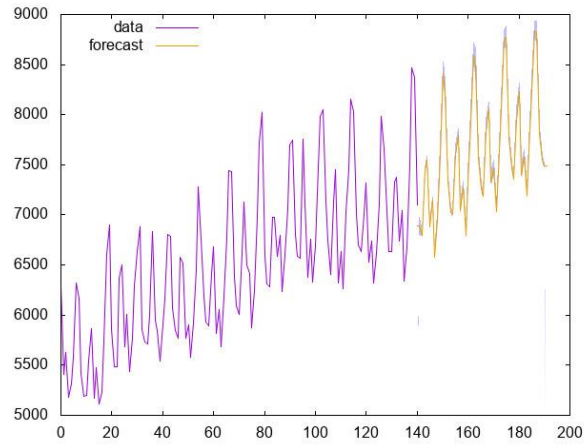
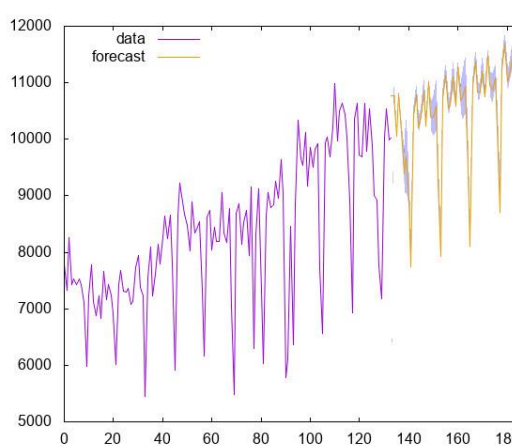
Auto-detect periodicity ...



Forecasting implies missing value imputation!

Wait, this is just a sine wave ... its easy ...

Maybe not...

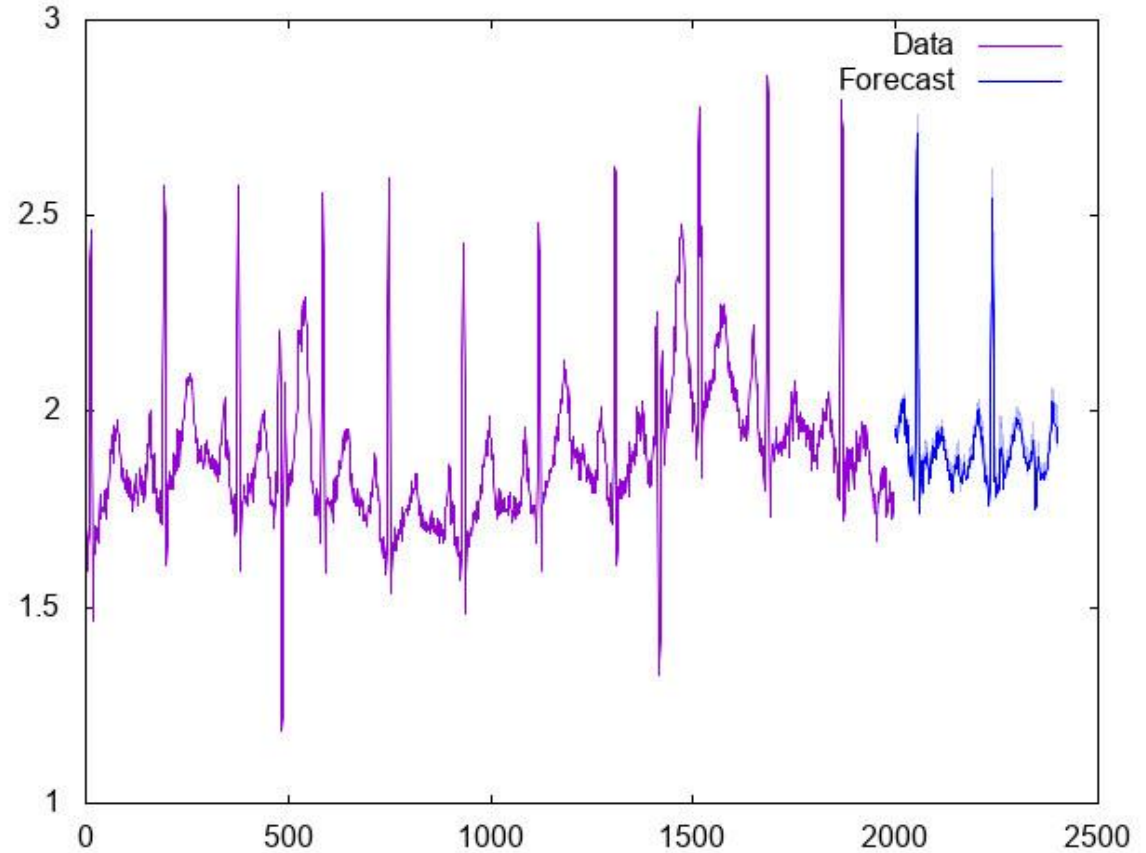


Realistic Data?

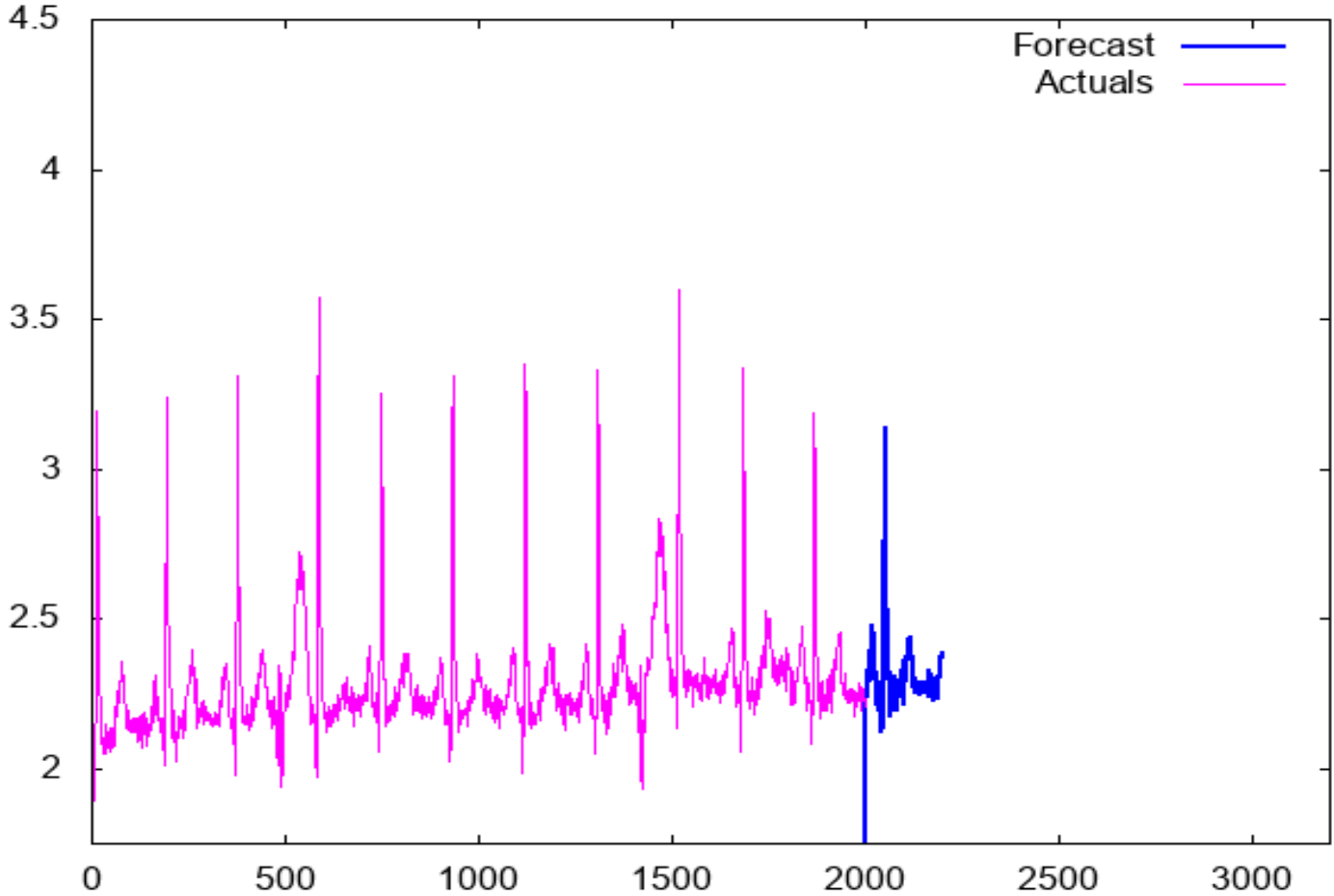
ECG (one lead)

Periodicity unclear ...

Shingle Size = 185



Test on hold out data



Paper in preparation!

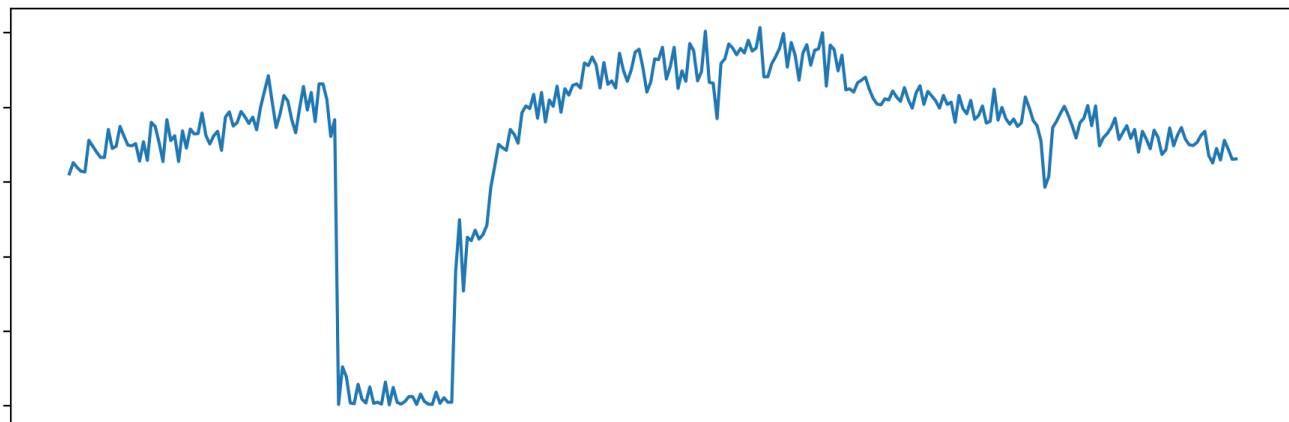
A New Explanation for Anomalies?

“This point is an anomaly because it is 3x the forecast”.

Makes it easier to set triggers/alerts.

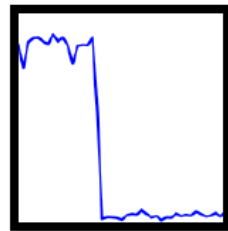
Semi-Supervised Learning on Data Streams

Amazon.com
Orders per
Minute

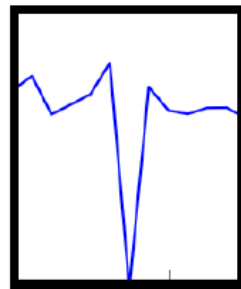


One Hour

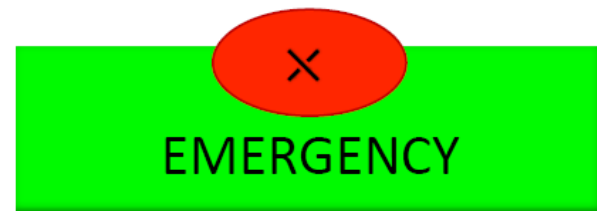
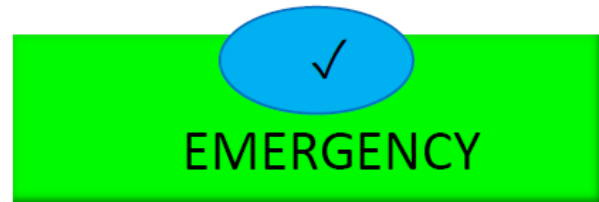




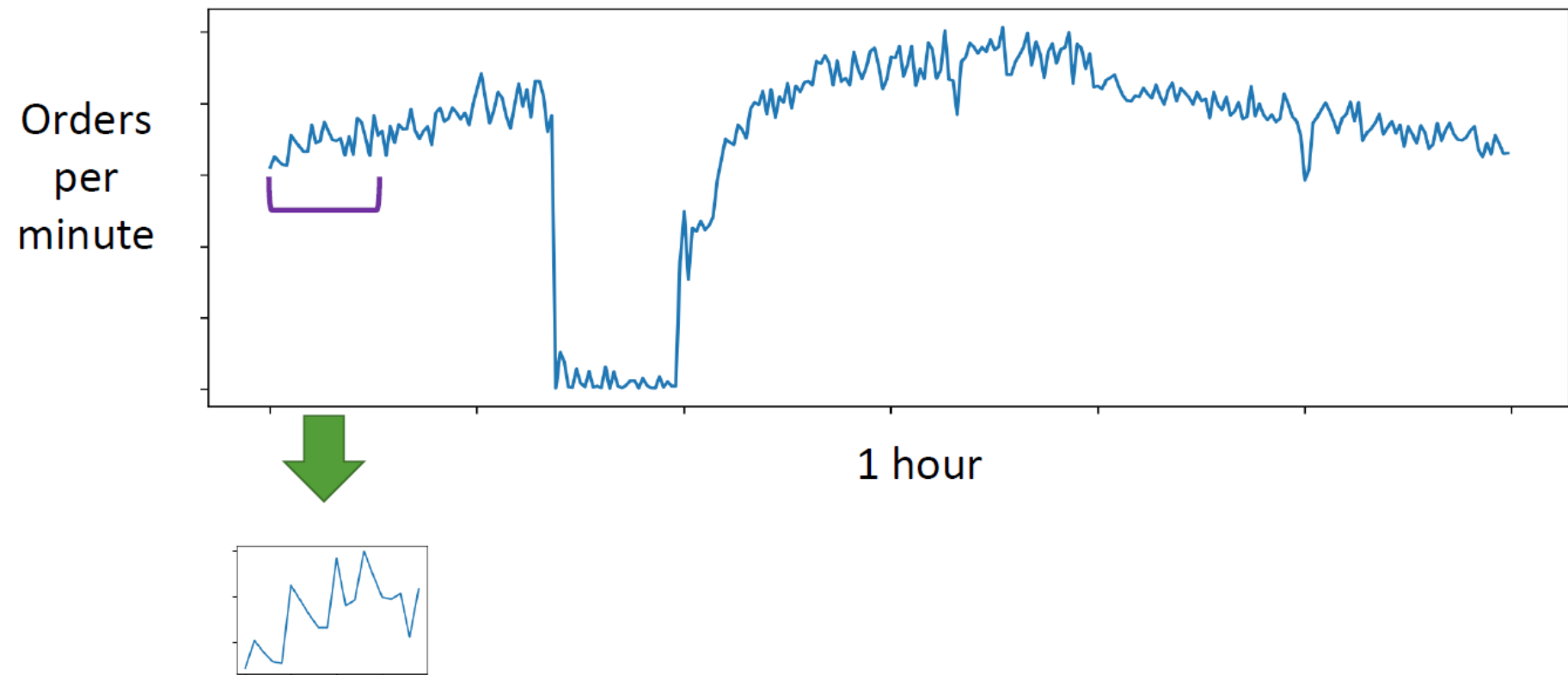
sustained dip



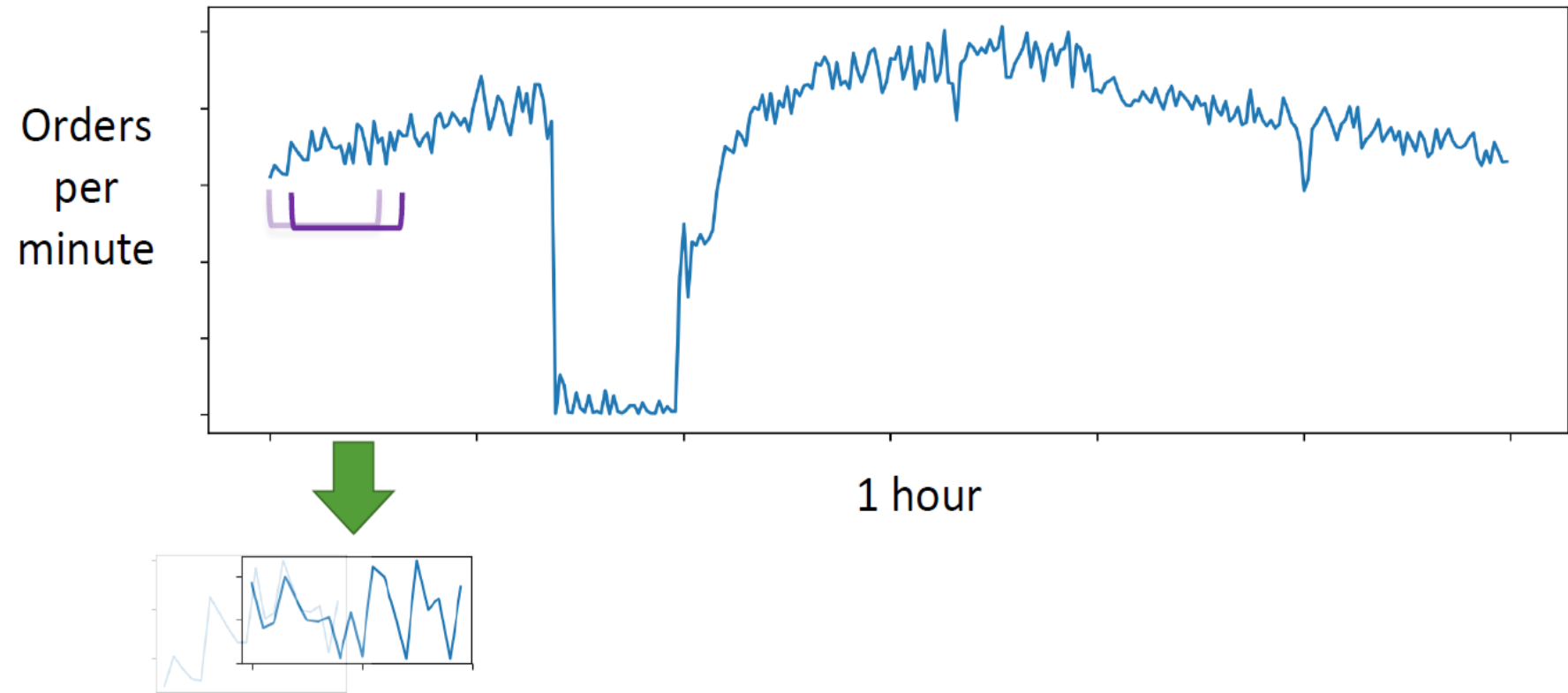
don't bother



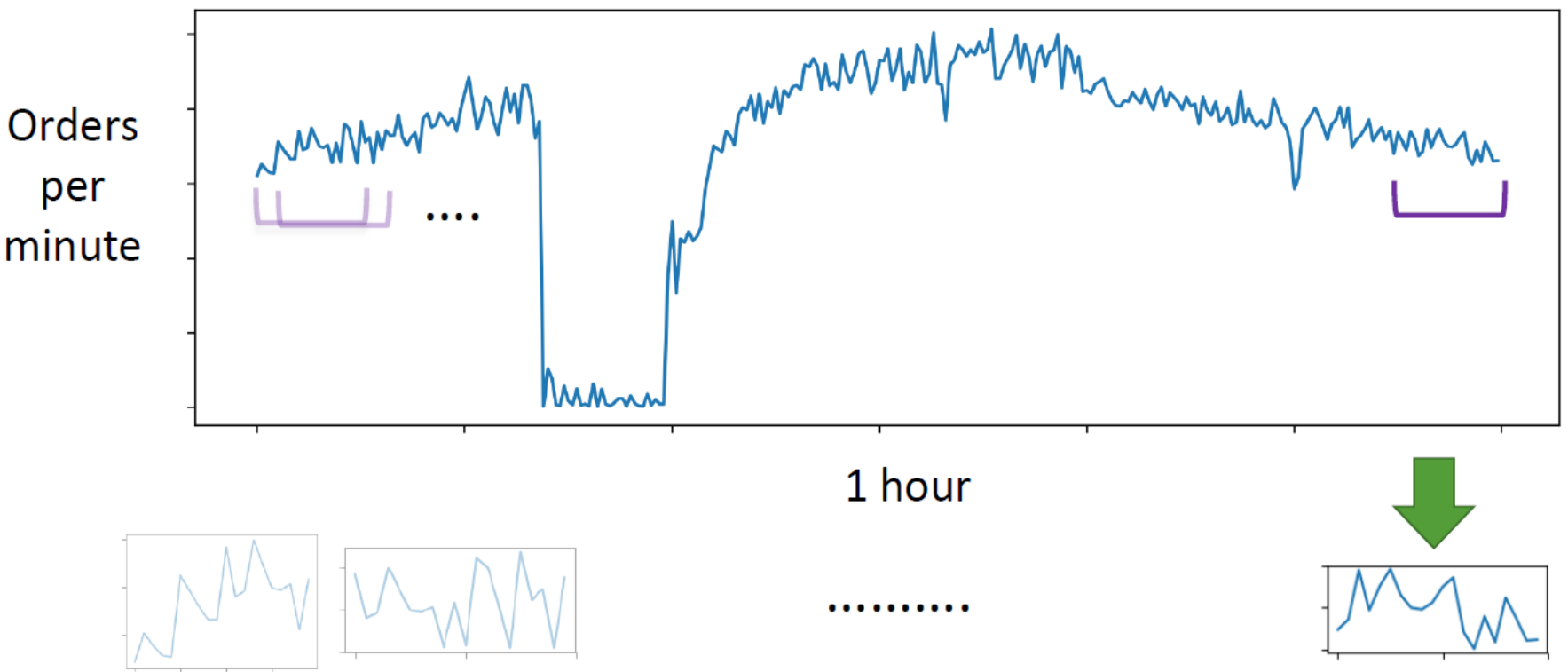
Can we incorporate user feedback
to make the system smarter over time?



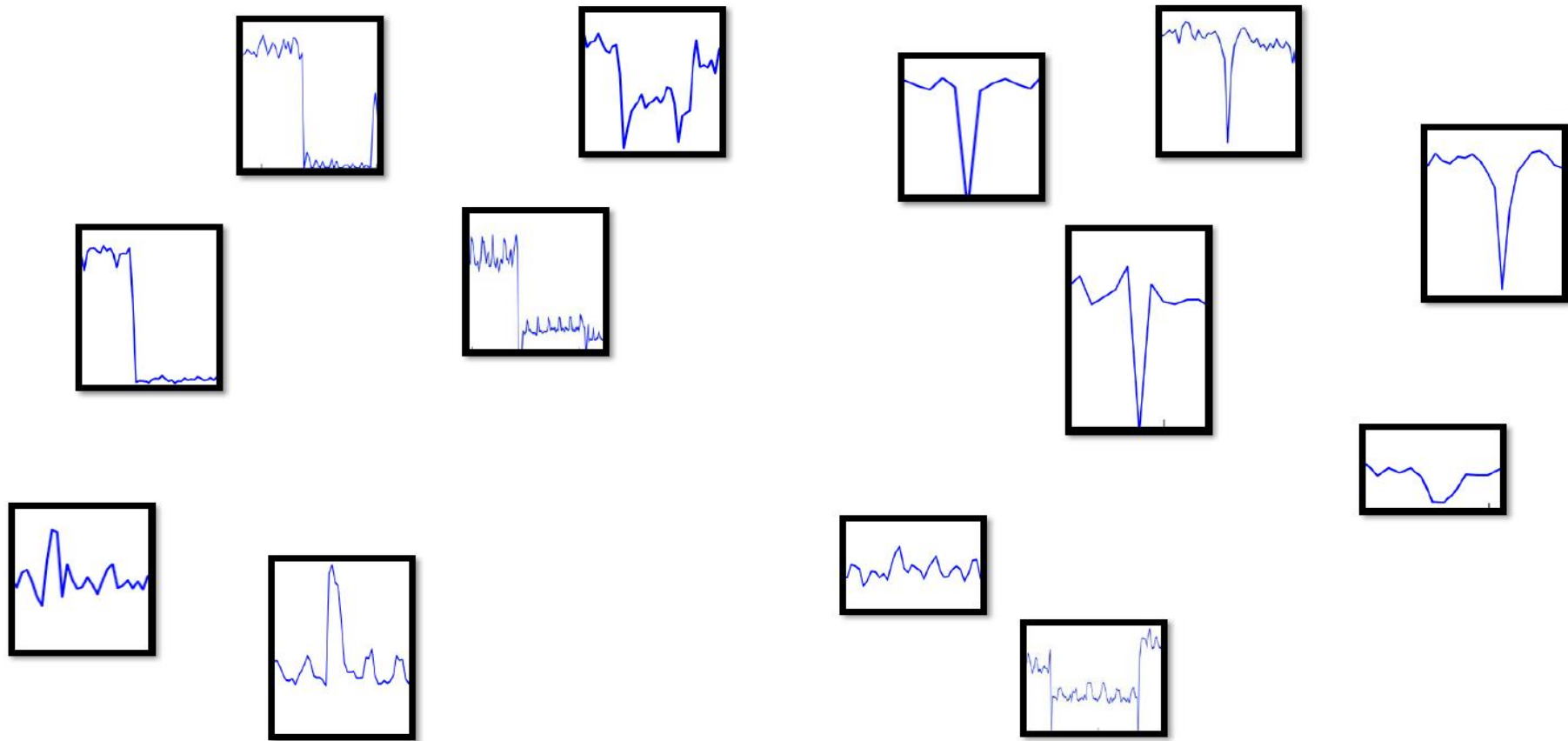
Step 1: (Pre-process) Fragment the data (shingling)



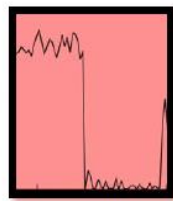
Step 1: (Pre-process) Fragment the data (shingling)



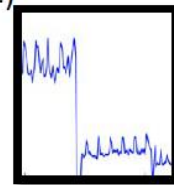
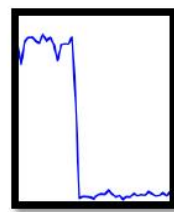
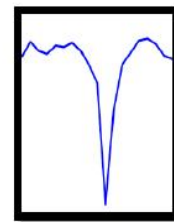
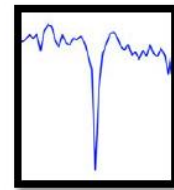
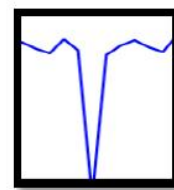
Step 1: (Pre-process) Fragment the data (shingling)



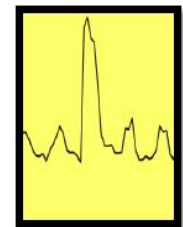
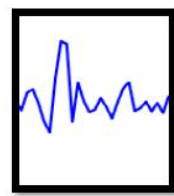
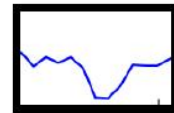
Step 2: Embed these fragments into a metric space



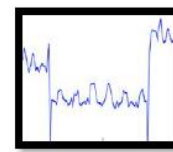
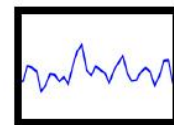
sustained dip (Sev-1)



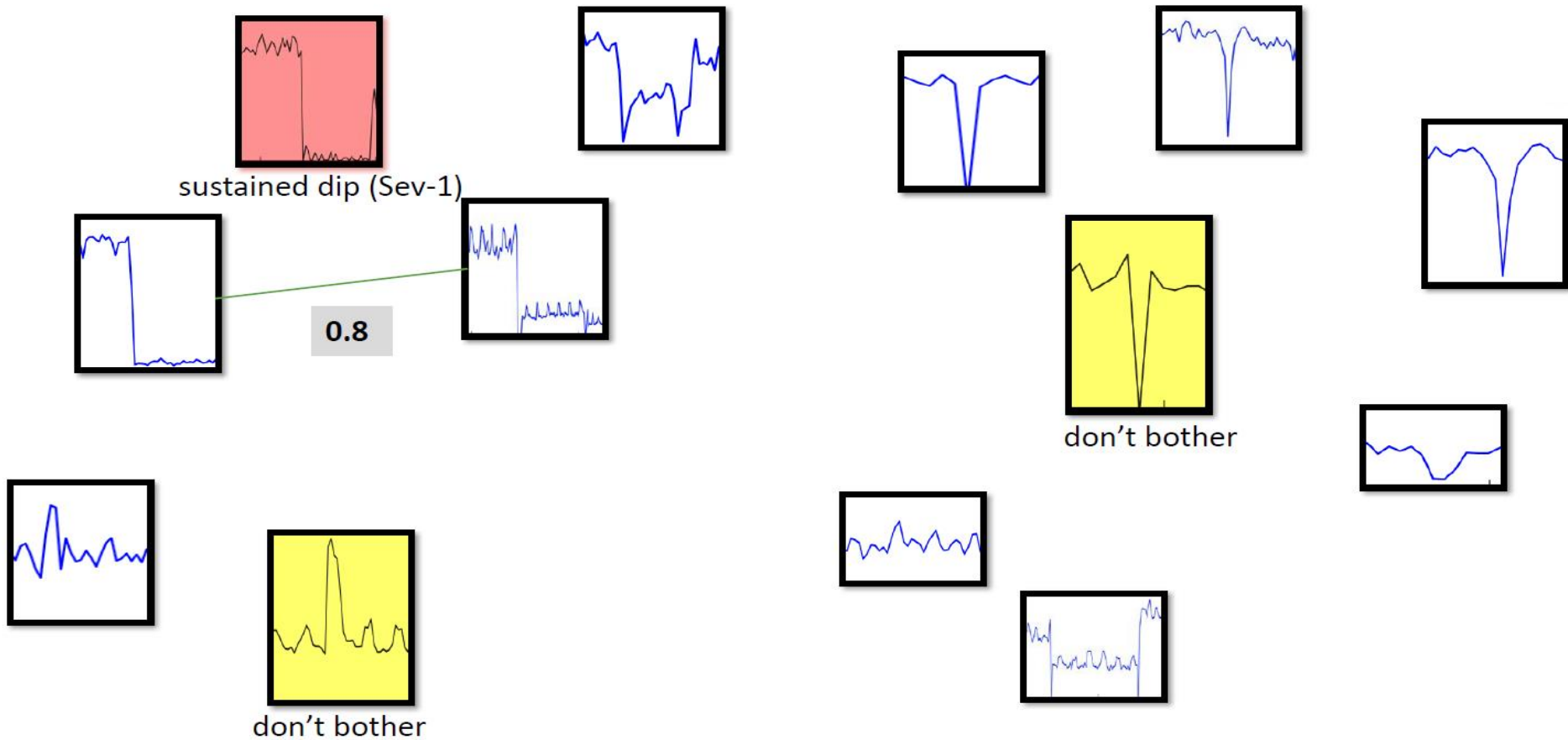
don't bother



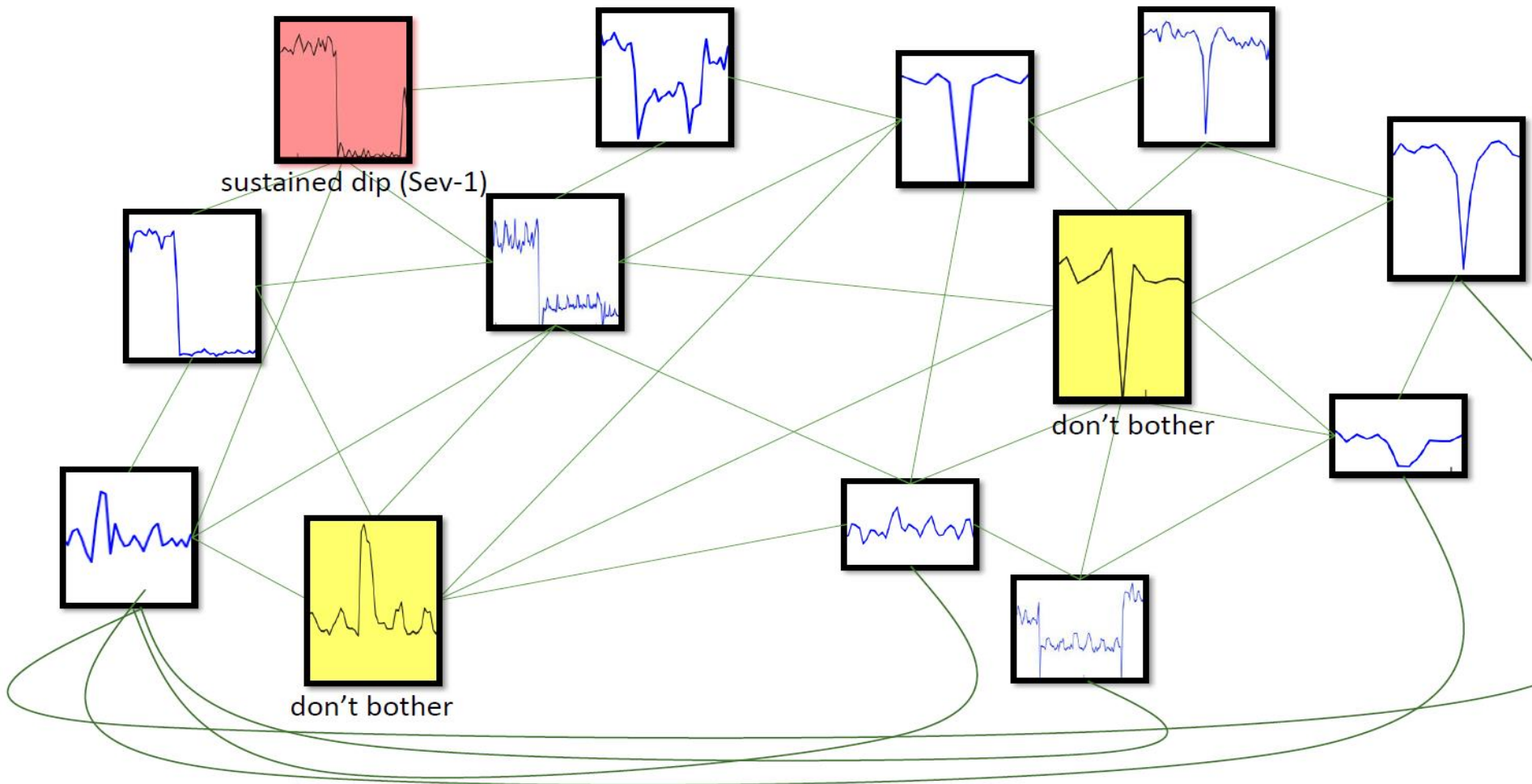
don't bother



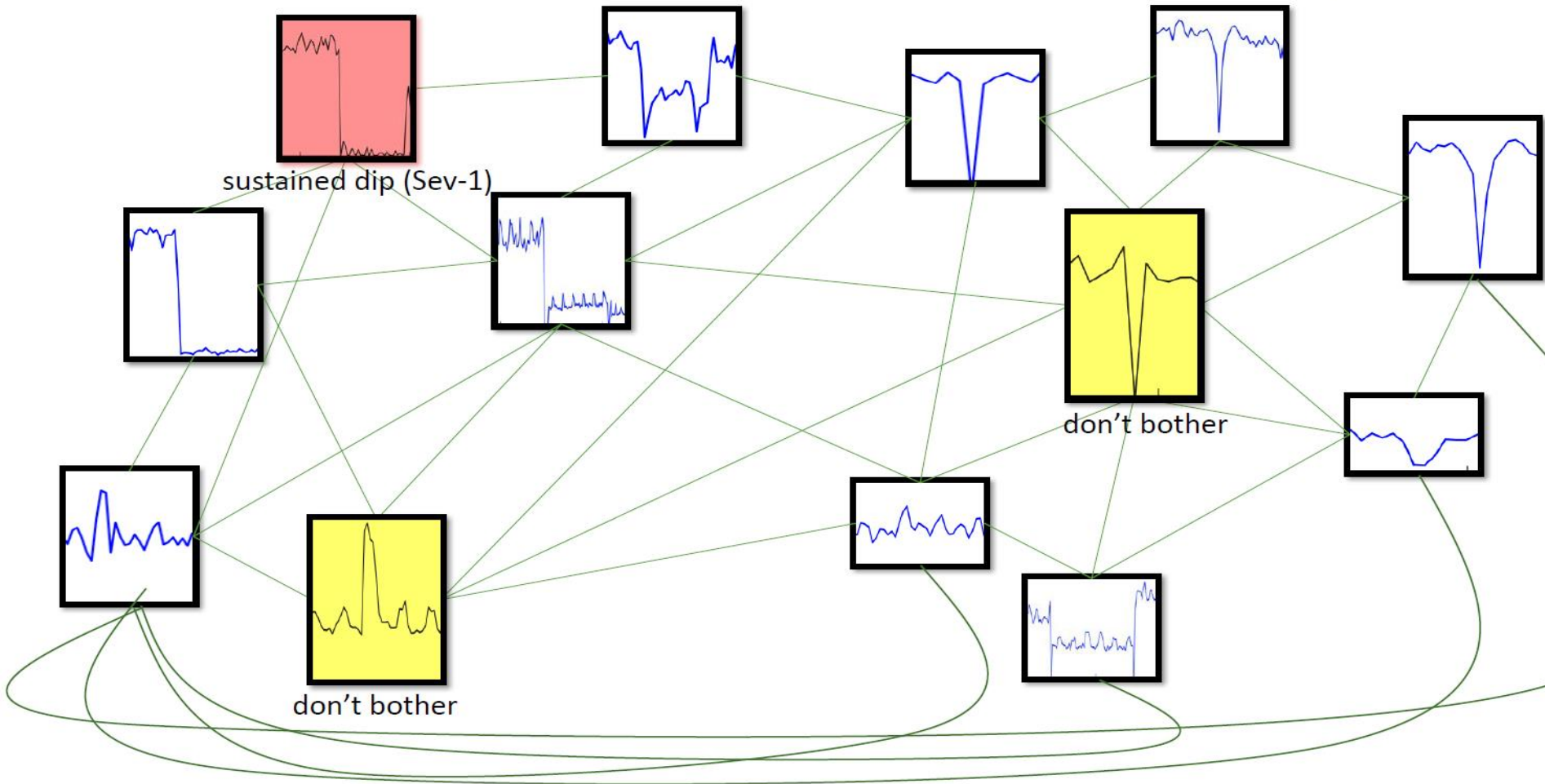
Expert provides few labels. Goal to “spread” this label to the remaining data.



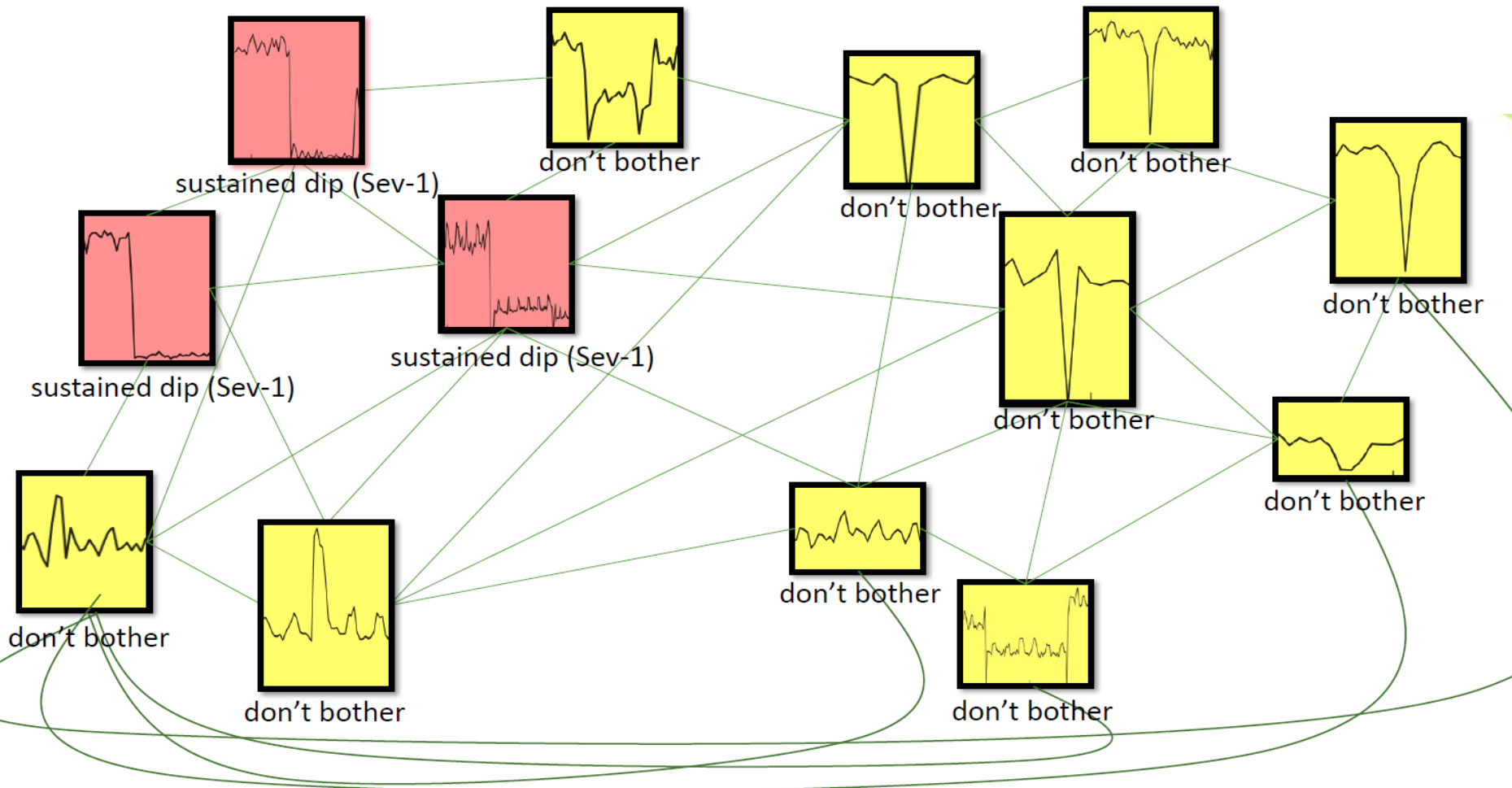
Step 3: Compute distances between fragments



Step 4: Construct a “similarity graph”

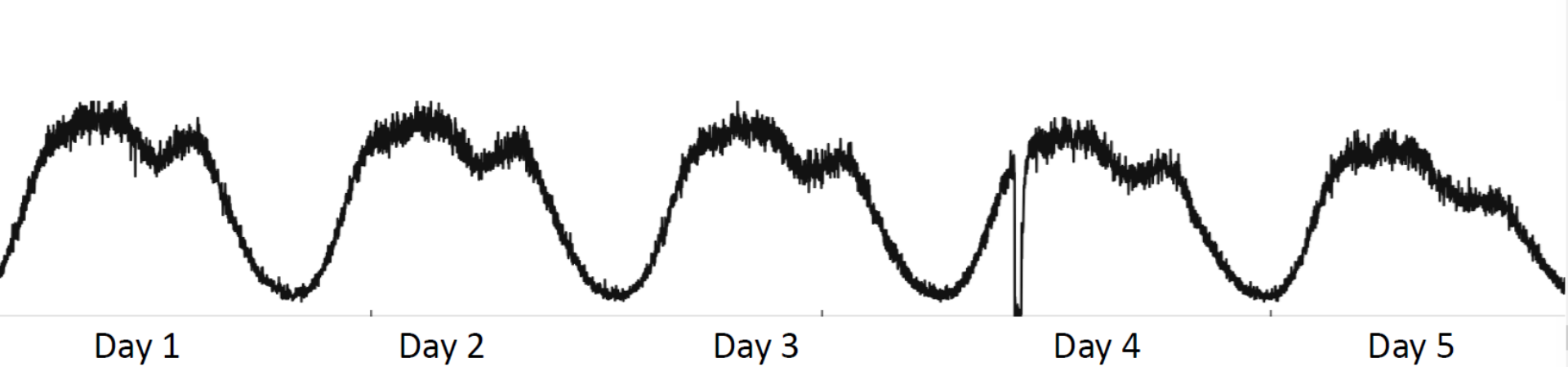


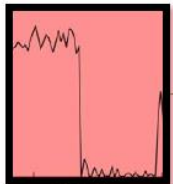
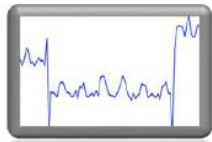
Step 5: "Label propagation" on this graph [Zhu Ghahramani Lafferty ICML '03]



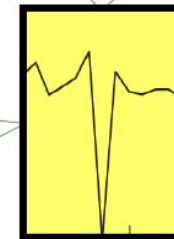
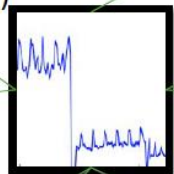
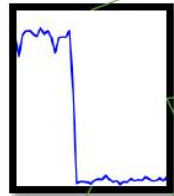
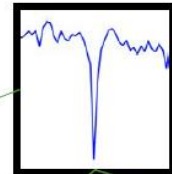
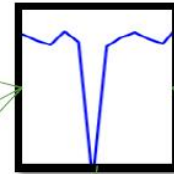
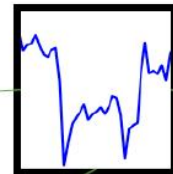
Output: Label on each unlabeled fragment

In a long stream...

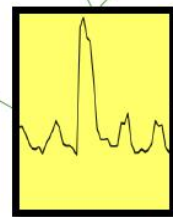
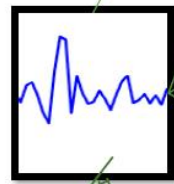
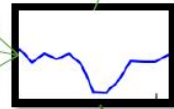
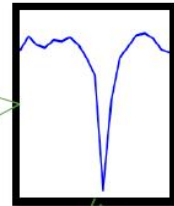




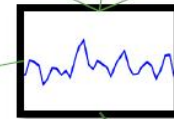
sustained dip (Sev-1)



don't bother



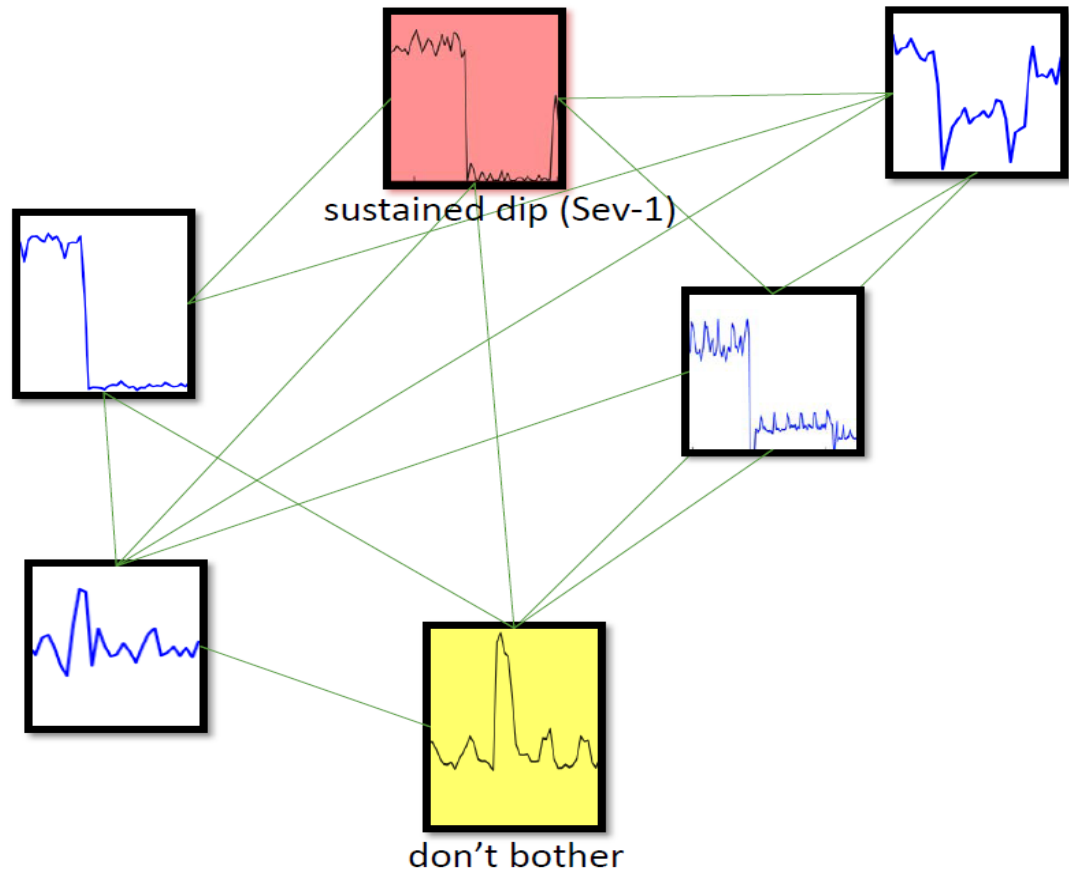
don't bother



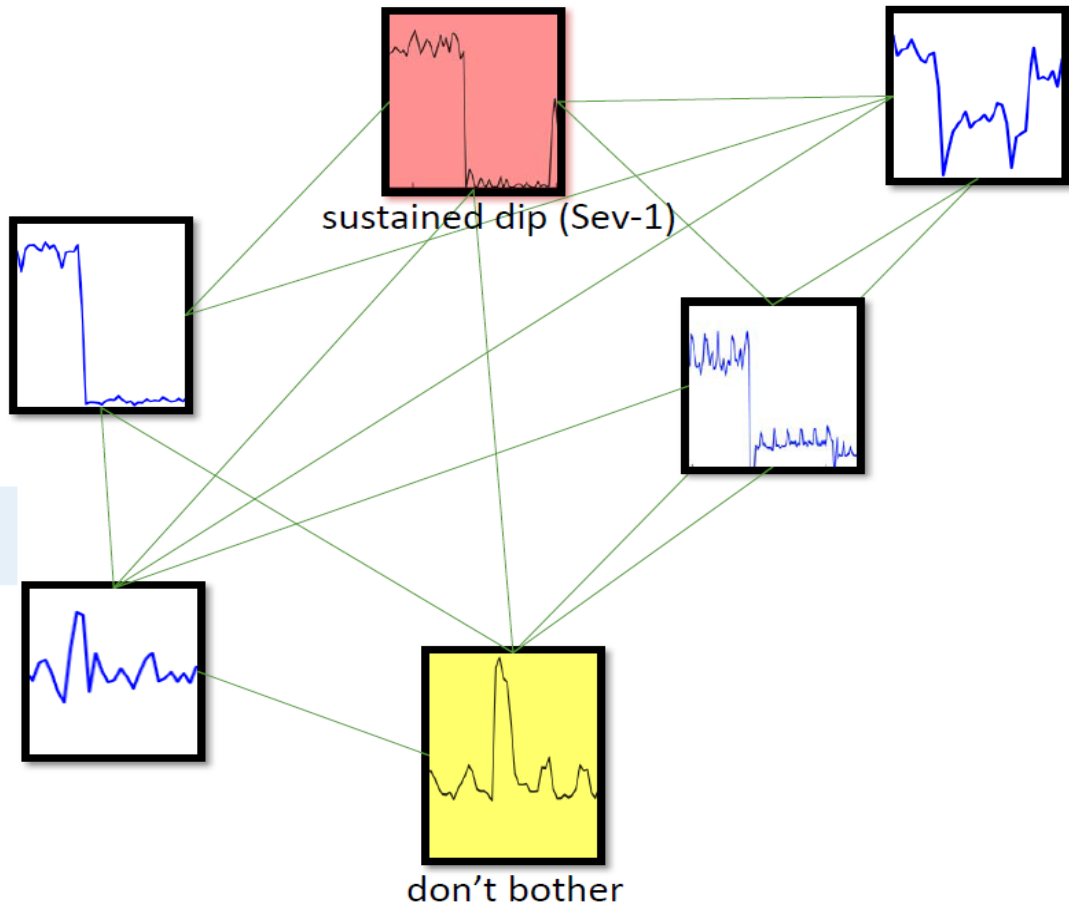
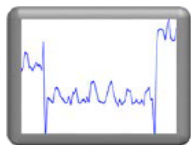
Fragments keep coming in

Our Algorithm in Pictures

Maintains the graph over a sliding window

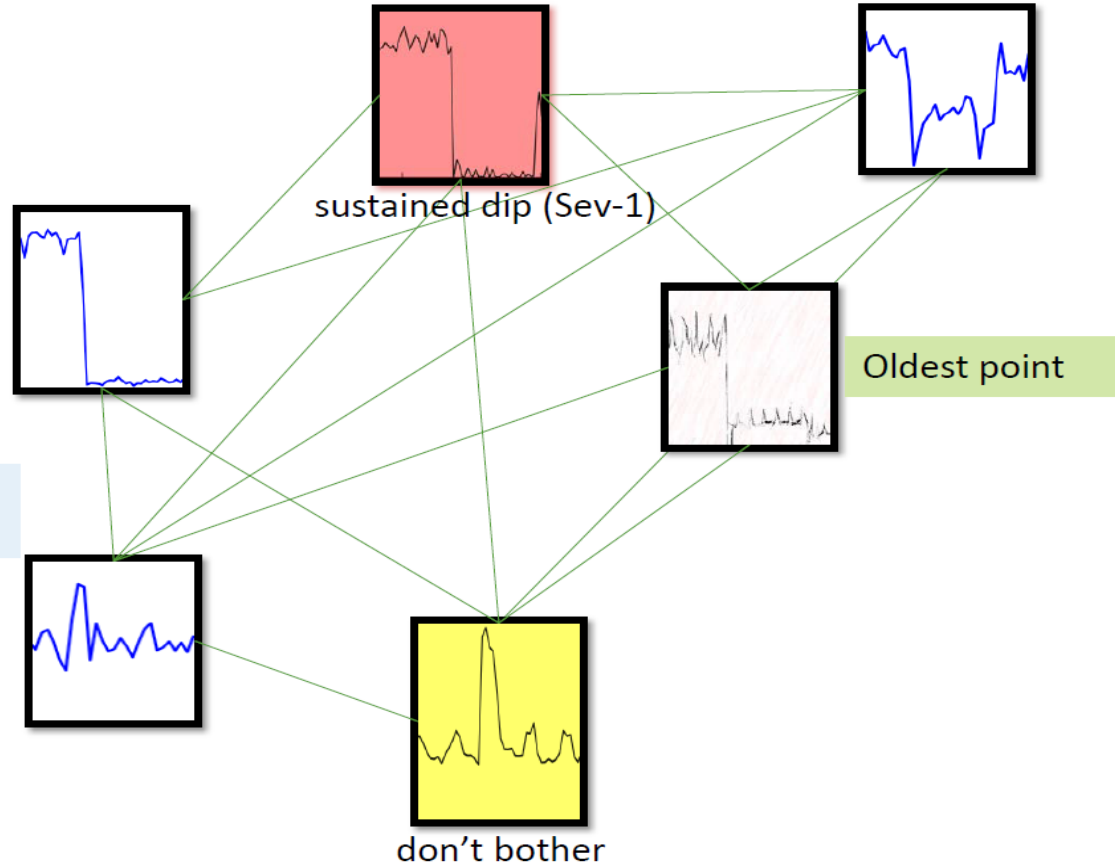
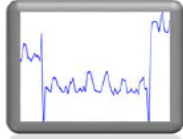


Our Algorithm in Pictures

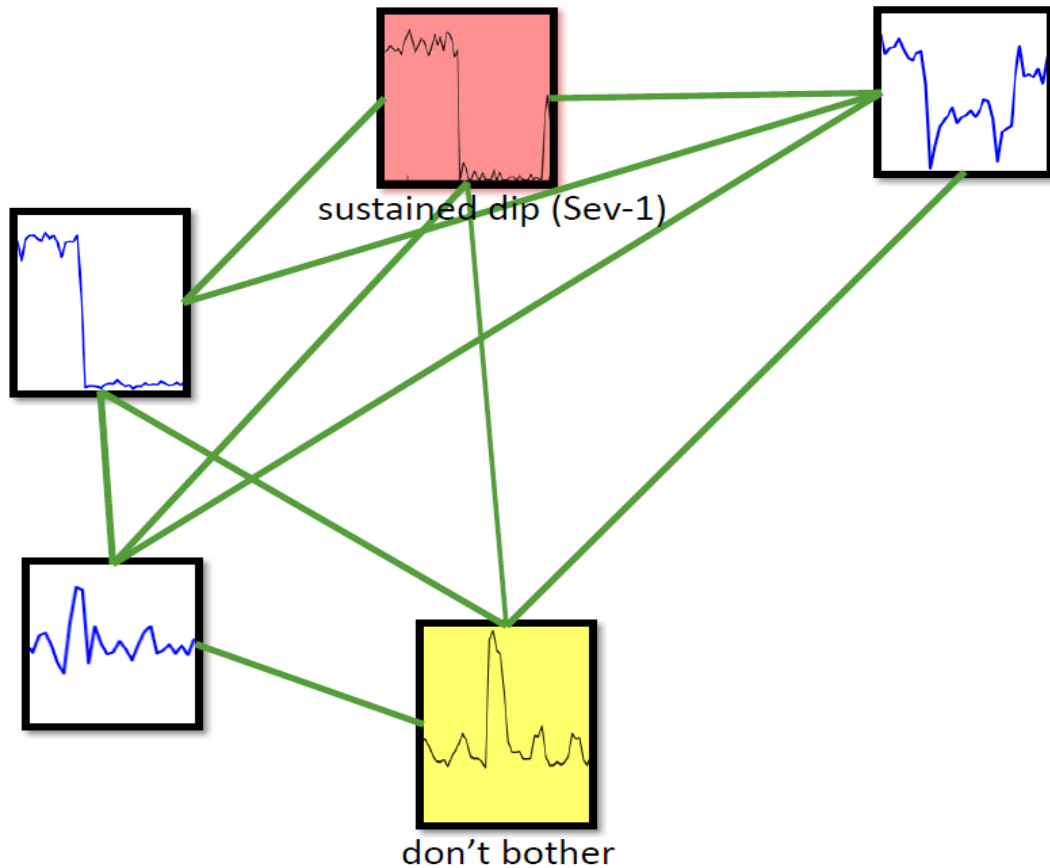
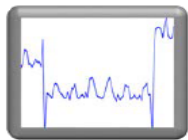


When a new point arrives

Our Algorithm in Pictures



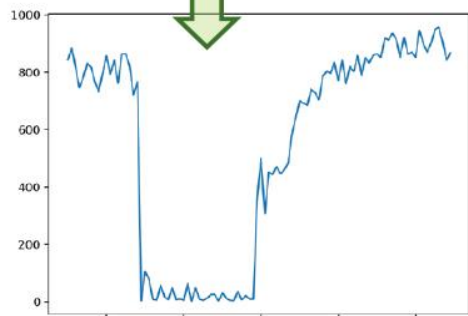
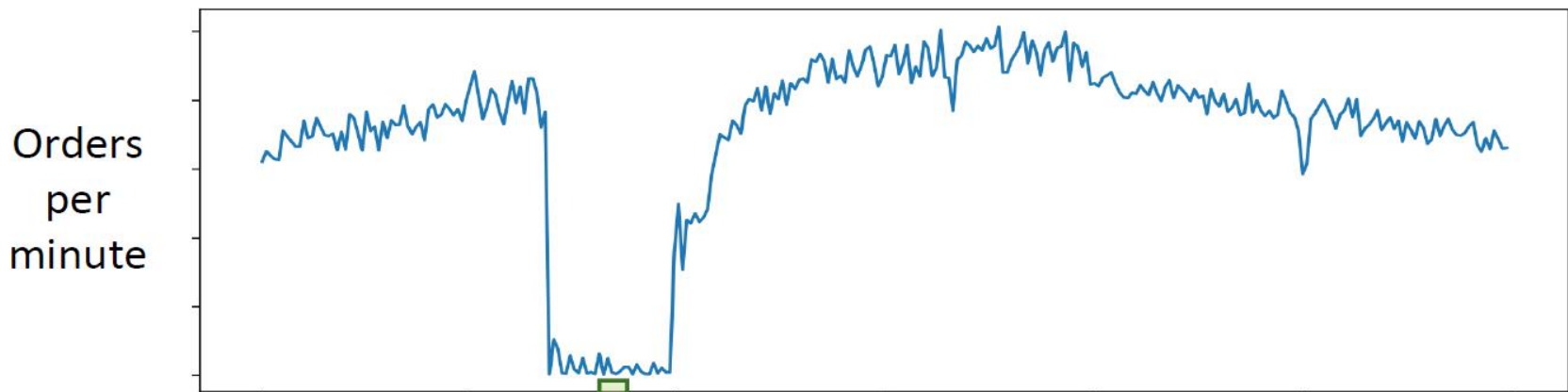
Our Algorithm in Pictures



Update weights
on the edges

by **star-mesh** transform
(a concept from electric circuits)*

Streaming Timeseries Classification



sustained dips

Amazon orders dataset

For 1 year of data: Accuracy > 95%
(just one labeled instance)
(memory consumed = 0.4 MB)

Proceedings of the 35th International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018.

Semi-Supervised Learning on Data Streams via Temporal Label Propagation

Tal Wagner¹ Sudipto Guha² Shiva Prasad Kasiviswanathan² Nina Mishra²

Abstract

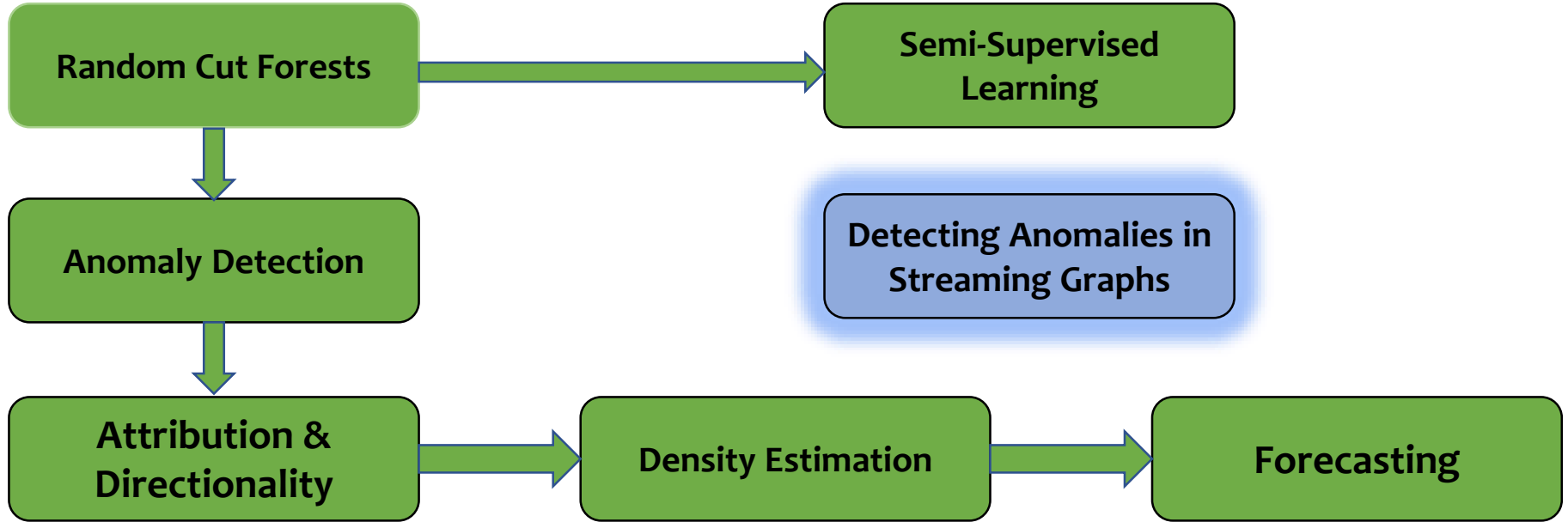
We consider the problem of labeling points on a fast-moving data stream when only a small number of labeled examples are available. In our setting, incoming points must be processed efficiently and the stream is too large to store in its entirety. We present a semi-supervised learning algorithm for this task. The algorithm maintains a small synopsis of the stream which can be quickly updated as new points arrive, and labels every incoming point by provably learning from the full history of the stream. Experiments on real datasets validate that the algorithm can quickly and accurately classify points on a stream with a small quantity of labeled examples.

ing of metrics arising from medical patient signals (ECG, EEG, fall detection), data centers (network, I/O and CPU utilization), or a camera mounted on a semi-autonomous car (for road conditions and obstacle detection). In these scenarios, unlabeled data is continuously streaming, but only a small number of manually labeled examples are provided – either at the beginning of the stream or as occasional user feedback. We want algorithms that leverage both inputs and learn how to classify stream elements, such as ECG arrhythmias, network intrusion alerts or driving conditions. Several other applications are given in (Goldberg et al., 2008), who defined a similar model, and in (Krempel et al., 2014).

In practice, this setting requires algorithms that run under severe time and memory constraints, since the labels are expected in real-time and the stream is generally too large to fully store in the memory. This poses a major challenge: How can we leverage the entire stream history to label a new point, when we can only store a tiny fraction of it?

1. Introduction

What we saw today...



SPOTLIGHT: Detecting Anomalies in Streaming Graphs

Dhivya Eswaran*, Christos Faloutsos*, Sudipto Guha†, Nina Mishra†

*Carnegie Mellon University †Amazon
{deswaran,christos}@cs.cmu.edu, {sudipto,nmishra}@amazon.com

ABSTRACT

How do we spot interesting events from e-mail/transportation logs? How can we detect port scan or denial of service attacks from IP-IP communication logs? In general, given a sequence of weighted, directed/bipartite graphs, each summarizing a snapshot of activity in a time window, how can we spot anomalous graphs containing the sudden appearance or disappearance of large dense subgraphs (e.g., near bicliques) in near real-time using sublinear memory? We propose a randomized sketching-based approach called SPOTLIGHT, which guarantees that an anomalous graph is mapped ‘far’ away from ‘normal’ instances in the sketch space with high probability for appropriate choice of parameters. Extensive experiments on real-world datasets show that SPOTLIGHT (a) improves accuracy by at least 8.4% compared to prior approaches, (b) is fast and can process millions of edges within a few minutes, (c) scales linearly with the number of edges and sketching dimensions and (d) leads to interesting discoveries in practice.

ACM Reference Format:

Dhivya Eswaran*, Christos Faloutsos*, Sudipto Guha†, Nina Mishra†. 2018. SPOTLIGHT: Detecting Anomalies in Streaming Graphs. In *Proceedings of ACM SIGKDD (SIGKDD '18)*. ACM, New York, NY, USA, 9 pages. https://doi.org/10.475/123_4

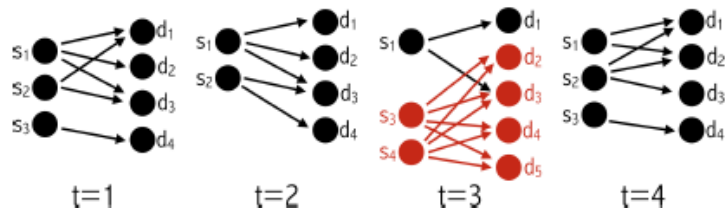


Figure 1: Sudden appearance of a dense subgraph at $t=3$.

attacks (port scan, denial of service) in network communication logs, interesting/fraudulent behavior creating spikes of activity in user-user communication logs (scammers who operate fast and in bulk), important events (holidays, large delays) creating abnormal traffic in/out flow to certain locations, etc. We are able to discover several of the above phenomena in real-world data (e.g., Fig. 2c).

We highlight two important aspects of the above definition. The (dis)appearance of a large dense subgraph is anomalous only if it is *sudden*, i.e., it has not been observed before or is not part of a slow evolution (e.g., steadily growing communities). Similarly, the sudden (dis)appearance of a large number of edges is anomalous only if the edges form a *dense* subgraph (the so-called *lockstep* behavior indicating fraud [5]). Fig. 1 illustrates this. In the evolution of a bipartite graph, e.g., user edits page, an anomalous dense directed subgraph appears at $t=3$, indicating a possible edit-war between

Contributors To This Project

Roger Barga, Dhivya Eswaran, Gaurav Ghare, Kapil Chhabra, Sudipto Guha, Shiva Kasiviswanathan, Nina Mishra, Gourav Roy, Joshua Tokle, and Tal Wagner