Anomaly Detection in Stream using RRCF

*Introducing RRCF as a fast and stable learner in stream data and detecting anomalies using it.*

Student:

Mohammad Doosti Lakhani

Instructor:

Dr. M. R. Kangavari

Course:

Machine Learning

# Introduction

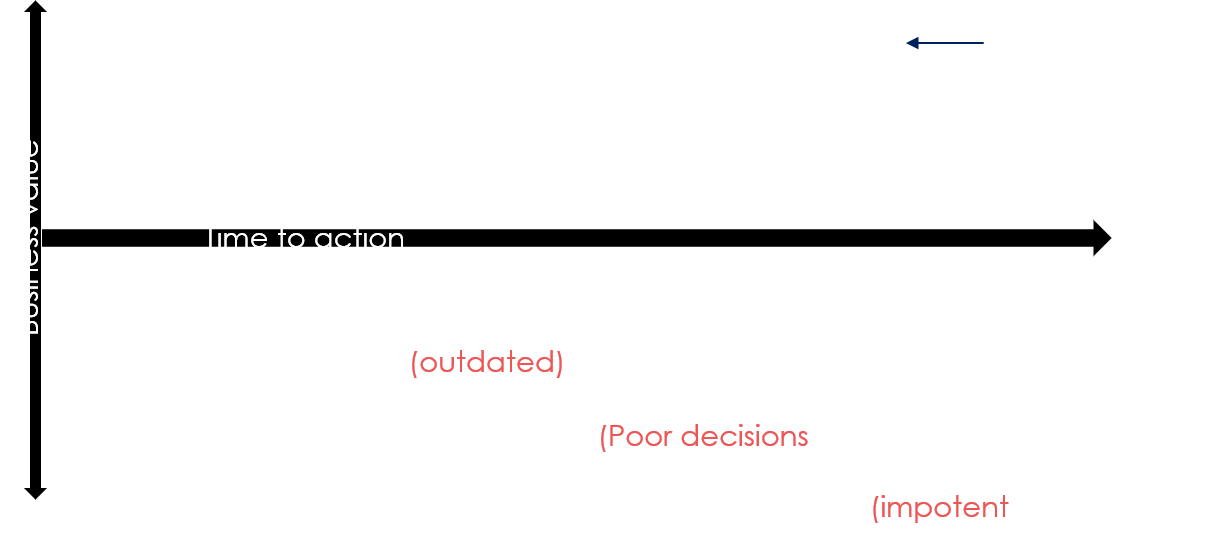
In this article we focus on the anomaly detection problem for dynamic data streams through the lens of random cut forests. Even though the problem has been well studied over the last few decades, the emerging explosion of data from the internet of things and sensors leads us to reconsider the problem. In most of these contexts the data is streaming and well-understood prior models do not exist.

But before going through concepts, let’s discuss main reason that it gained attention. It can be argued in two sections:

1. The importance of stream data
2. Main challenge of stream data

These questions can be answered in many different ways but in this area but we can distinguish main reasons in this way:

1. Importance of stream data:
   1. All data originates in real time e.g. image segmentation, language models
   2. Emerging explosion of IoT
   3. It exists in everyday tasks
2. Main challenge of stream data:
   1. Batch operations take too long
   2. Insights are perishable



So, based on the aforementioned issues, a algorithm that can digest data as it is generated, process it on the fly and does real time machine learning is desirable. To address these issues, we define following questions:

Two central questions in this regard are:

1. how do we deﬁne anomalies?
2. what data structure do we use to efﬁciently detect anomalies over dynamic data streams?

In this report, this will be tackled.

# Definitions

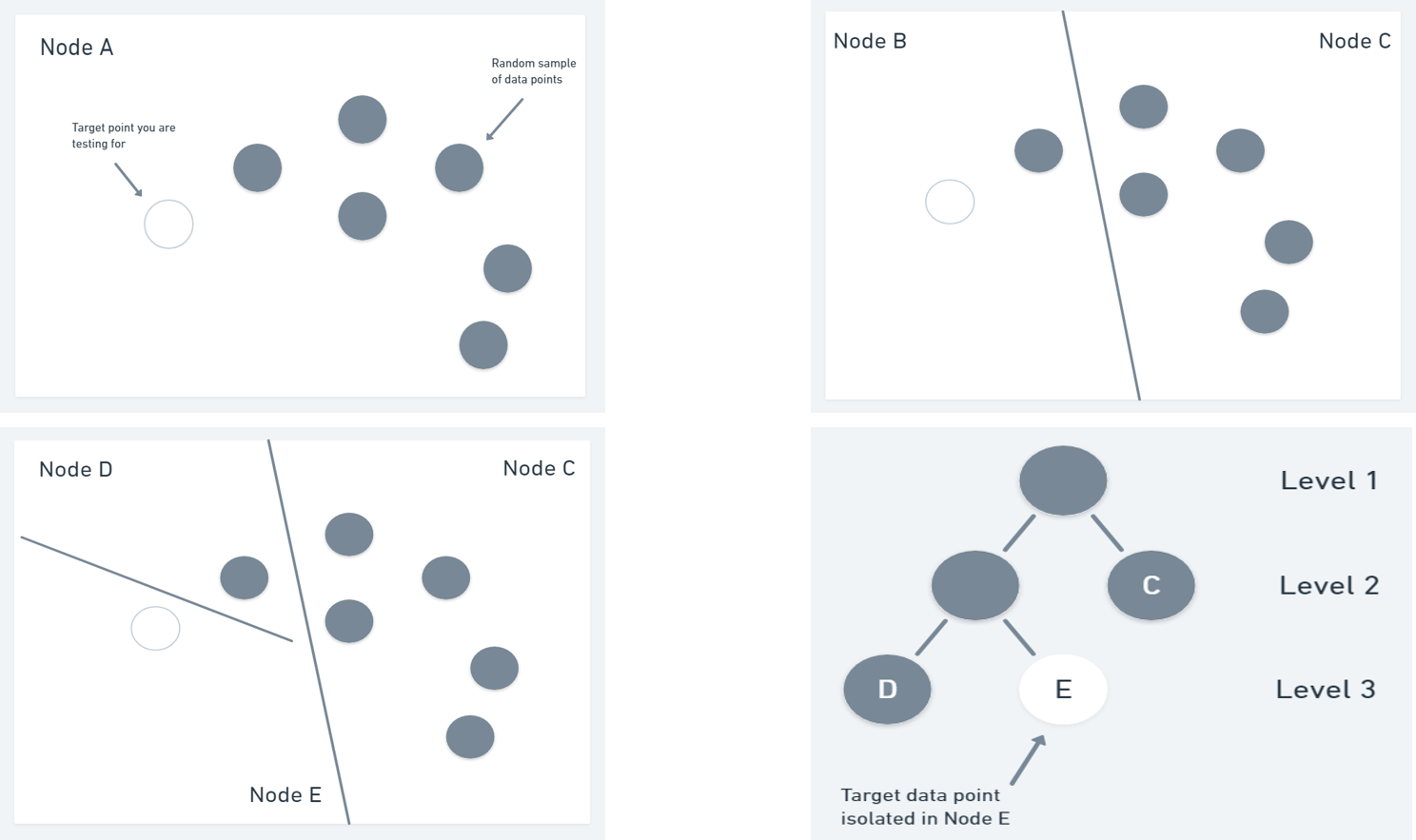
Let’s first answer the two aforementioned questions in a precise manner to have some clue about the ideas.

|  |  |  |
| --- | --- | --- |
| For first question, the problem has been viewed from the perspective of model complexity and say that a point is an anomaly if the complexity of the model increases substantially with the inclusion of the point. |  | For question two, randomized approaches have been considered as they are strong mainly in supervised fashion. But most of the algorithms based on random trees have been studied well for anomalies in a stream data situation. So, we deal with issue by proposing Robust Random Cut Tree. |

## Definition 1: RRCT

RRCT is an unsupervised but very fast type of decision tree. In below, we can see the main idea:

* Choose a random dimension proportional to:  
  where
* Choose
* Let and and recurse on and

To understand how it works, an example can be expressed which as follows:  
  


To explain this image, on top left, we start by mentioning that we want to isolate that particular white node. To do so, we use random cuts on dimensions of the data so get top right image, and we continue doing until we isolate white note which we can see it as a node in the corresponding constructed tree in bottom right image.

This is the core of this article as an anomaly point can be expressed by its distance from root. Nodes that in the original data are far from the data points (outliers/anomalies) are still far but from root of tree. So, based on this idea, we may need to define anomaly once more.

## Definition 2: Anomaly in RCF

Here are the major attributes of anomaly point that can be expressed:

* Anomaly points are isolated much faster and they will be on top of the tree
* The points near the root will get higher score
* The score is distance based, so the points far from normal clusters get higher score (nearer to root)
  + So, a point will be anomaly if it increases the size of tree profoundly
* Same idea has been used for test
  + If test node is near to root, then it is probably an anomaly
* If a point is far in from data in N-dim data, it will be as far relatively in RCF

# Theorems

## Theorem 1:

***Consider the algorithm in Deﬁnition 1. Let the weight of a node in a tree be the corresponding sum of dimensions Given two points the tree distance between u and v to be the weight of the least common ancestor of u, v. Then the tree distance is always at least the Manhattan distance and in expectation, at most times .***

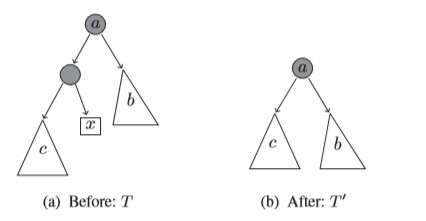
The theorem is interesting because it implies that if a point is far from others (as is the case with anomalies) that it will continue to be at least as far in a random cut tree in expectation. The proof of Theorem 1 follows along the same lines of the proof of approximating ﬁnite metric spaces by a collection of trees (Charikaretal.,1998).

Thus, the RRCF ensemble contains sufﬁcient information that allows us to determine distance based anomalies, without focusing on the speciﬁcs of the distance function. Moreover, the distance scales are adjusted appropriately based on the empty spaces between the points since the two bounding boxes may shrink after the cut.

## Theorem 2:

***Given a tree drawn according to ; if we delete the node containing the isolated point x and its parent (adjusting the grandparent accordingly), then the resulting tree has the same probability as if being drawn from . Likewise, we can produce a tree as if drawn at random from is time which is times the maximum depth of T, which is typically sublinear in .***

Theorem 2 demonstrates an intuitively natural behavior when points are deleted. In effect, if we insert x, perform a few more operations and then delete x, then not only do we preserve distributions but the trees remain very close to each other — as if the insertion never happened. This behavior is a classic desideratum of sketching algorithms. In the below figure, we can see an intuitive demonstration of this theorem:



## Theorem 3:

***We can maintain a random tree over a sample even as the sample is updated dynamically for streaming data using sublinear update time and space.***

We can now use reservoir sampling (Vitter, 1985) to maintain a uniform random sample of size or a recency biased weighted random sample of size (Efraimidis & Spirakis, 2006), in space proportional to on the ﬂy. In effect, the random sampling process is now orthogonal from the robust random cut forest construction

## Theorem 4:

***Given a tree for sample , if there exists a procedure that downsamples via deletion, then we have an algorithm that simultaneously provides us a downsampled tree for every downsampling rate.***

Theorems 3 and 4 taken together separate the notion of sampling from the analysis task and therefore eliminates the need to ﬁnetune the sample size as an initial parameter. Moreover, the dynamic maintenance of trees in Theorem 3 provides a mechanism to answer counterfactual questions as given in Theorem 5.

## Theorem 5:

***Given a tree for sample , and a point we can efﬁciently compute a random tree in , and therefore answer questions such as: what would have been the expected depth had been included in the sample.***

Intuitively, we label a point as an anomaly when the joint distribution of including the point is signiﬁcantly different from the distribution that excludes it. Theorem 5 allows us to efﬁciently (pretend) sketch the joint distribution including the point .

# Maintenance on a Stream

In this section we discuss how Robust Random Cut Trees can be dynamically maintained. In the following, let RRCF(S) be a distribution over trees by running Definition 1 on S. Consider the following operations.

|  |  |  |
| --- | --- | --- |
| Deletion |  | Insertion |
| Given drawn from distribution and produce a drawn from .  Algorithm:    If were drawn from the distribution then Algorithm 1 produces a tree which is drawn at random from the probability distribution  .  We can derive inverse concept in the same manner for insertion. |  | *Definition: Given drawn from distribution and*  *produce a drawn  from .*  Algorithm: |

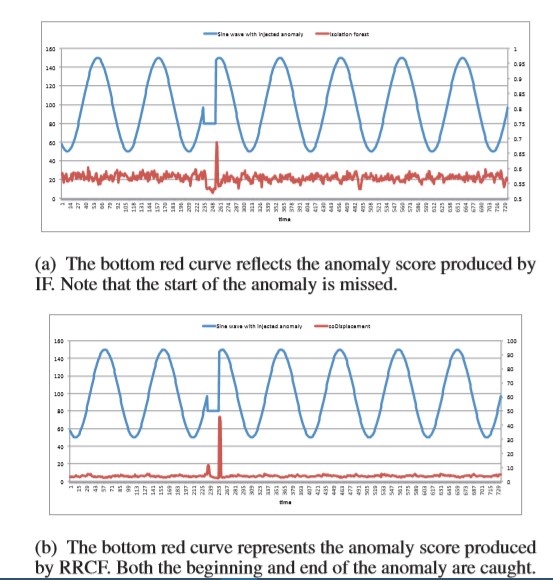
# Applications

## Real Life Data: NYC Taxi cabs

Authors considered data as a stream of the total number of passengers aggregated over a 30 minute time window. Data is collected over a 7month time period. Note while this is a 1-dimensional datasets, authors treat it as a 48-dimensional data set where each point in the stream is represented by a sliding window.   
The famous holidays have been manually labeled as anomaly. For simplicity, authors label a 30 minute window an anomaly if it overlaps one of these days.

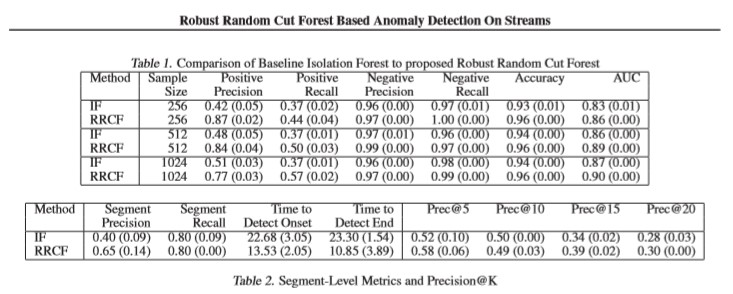
Authors treat the data as a stream – after observing points , our goal is to score the st point. The score that we produce for is based only on the previous data points , but not their labels. They use as the baseline.

In below images, we can see the behavior of these two algorithms regarding a anomaly window:



Also, in below image, we can see the differences between and model w.r.t. multiple metrics for a better comparison. The training set contains all points before time and the test set all points after time . The threshold is chosen to optimize the measure (harmonic mean of precision and recall). Authors focus their attention on positive precision and positive recall.

For the ﬁner granularity data in the taxi cab data set, we view the ground truth as segments of time when the data is in an anomalous state. Our goal is to quickly and reliably identify these segments.



# References

1. Guha, Sudipto, et al. "Robust random cut forest based anomaly detection on streams." International conference on machine learning. 2016
2. Wagner, Tal, et al. "Semi-supervised learning on data streams via temporal label propagation." International Conference on Machine Learning. 2018
3. Eswaran, Dhivya, et al. "Spotlight: Detecting anomalies in streaming graphs." Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. 2018
4. Some of images are barrowed from [Manning Publications](https://freecontent.manning.com/the-randomcutforest-algorithm/)