Anomaly Detection

Isolation Forest

Andrei Hîji

Computer Science Department

Faculty of Mathematics and Computer Science

University of Bucharest

Email: `andrei-iulian.hiji@unibuc.ro`

Outline

- Isolation Forest
- Extended Isolation Forest
- Generalized Isolation Forest
- Deep Isolation Forest

The course references are the initial paper for IForest by Liu, Ting, and Zhou 2008 and its variations: Extended Isolation Forest Hariri, Kind, and Brunner 2019, Generalized Isolation Forest Lesouple et al. 2021, Deep Isolation Forest Xu et al. 2023

Normal instances profiling vs explicit anomaly isolation

- Most methods try to profile normal instances and then identify those instances that don't comply with the resulted profile
  - not optimized for anomaly detection
  - existing methods have high computational complexity (constraints on data size and dimensionality)

Normal instances profiling vs explicit anomaly isolation

- Most methods try to profile normal instances and then identify those instances that don't comply with the resulted profile
  - not optimized for anomaly detection
  - existing methods have high computational complexity (constraints on data size and dimensionality)
- Explicit isolation exploits 2 properties of anomalies
  - fewer instances (minority)
  - very different attributes (compared to normal instances)

Anomaly isolation - basic idea

A binary tree - used to isolate every instance by recursively partitioning the data space with random binary splits

- root node will contain initial data
- each binary split will create 2 child nodes
- stop when each instance is isolated

Assumption: anomalies will be isolated faster (closer to the root of the tree because they are 'few and different')

Isolation Tree (iTree)

Definition

Let T be a node of an isolation tree. T can be an external-node with no child, or an internal-node with one test and exactly two child nodes (T_l, T_r). A test consists of an attribute q and a split value p such that the test $q < p$ divides data points into T_l and T_r

Isolation Tree (iTree)

Definition

Let T be a node of an isolation tree. T can be an external-node with no child, or an internal-node with one test and exactly two child nodes (T_l, T_r). A test consists of an attribute q and a split value p such that the test $q < p$ divides data points into T_l and T_r

iTree properties:

- proper binary tree (each node has 0 or 2 child nodes)
- a fully grown iTree has n external nodes (where n-number of instances)
- a fully grown iTree has $n - 1$ internal nodes

How to build an iTree?

Let $X = \{x_1, x_2, ..x_n\}$ - a sample of n instances where $x_i \in \mathbb{R}^d$

How to build an iTree?

Let $X = \{x_1, x_2, ..x_n\}$ - a sample of n instances where $x_i \in \mathbb{R}^d$

1. Randomly select an attribute q

How to build an iTree?

Let $X = \{x_1, x_2, ..x_n\}$ - a sample of n instances where $x_i \in \mathbb{R}^d$

1. Randomly select an attribute q
2. Randomly choose a split value p

How to build an iTree?

Let $X = \{x_1, x_2, ..x_n\}$ - a sample of n instances where $x_i \in \mathbb{R}^d$

1. Randomly select an attribute q
2. Randomly choose a split value p
3. Using the test $q < p$ divide X in X_l and X_r

How to build an iTree?

Let $X = \{x_1, x_2, ..x_n\}$ - a sample of n instances where $x_i \in \mathbb{R}^d$

1. Randomly select an attribute q
2. Randomly choose a split value p
3. Using the test $q < p$ divide X in X_l and X_r
4. Recursively partition X_l and X_r using steps 1, 2, 3 until:
   - the iTree reaches a height limit
   - $|X| = 1$
   - all instances in X have the same values

Normal point vs anomaly isolation

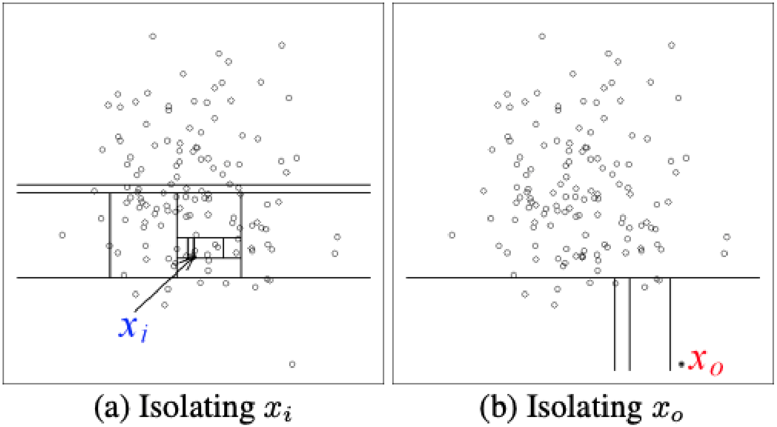


Figure: Isolation process (Liu, Ting, and Zhou 2008)

- 12 random partitions needed to isolate x_i; only 4 needed for x_o

Anomaly score derivation

Definition

Path Length $h(x)$ of a point x represents the number of edges x traverses in an iTree from the root node until the traversal is terminated at an external node.

Why **Path length** can't be used as an anomaly score?

Anomaly score derivation

Definition

Path Length $h(x)$ of a point x represents the number of edges x traverses in an iTree from the root node until the traversal is terminated at an external node.

Why **Path length** can't be used as an anomaly score?

We need a bounded interval, independent of number of instances n

Anomaly score derivation

Properties of **path length**:

- maximum path length grows in the order of n

Anomaly score derivation

Properties of **path length**:

- maximum path length grows in the order of n
- average path length grows in the order of $log(n)$

Anomaly score derivation

Properties of **path length**:

- maximum path length grows in the order of n
- average path length grows in the order of $log(n)$

Equivalence of iTrees with Binary Search Trees:

- both are proper binary trees

Anomaly score derivation

Properties of **path length**:

- maximum path length grows in the order of n
- average path length grows in the order of $log(n)$

Equivalence of iTrees with Binary Search Trees:

- both are proper binary trees
- a path search ending in an external node is equivalent with an unsuccessful search in a BST

Anomaly score derivation

Properties of **path length**:

- maximum path length grows in the order of n
- average path length grows in the order of $log(n)$

Equivalence of iTrees with Binary Search Trees:

- both are proper binary trees
- a path search ending in an external node is equivalent with an unsuccessful search in a BST
- average path length of an iTree = average path length of unsuccessful search in BST

Anomaly score derivation

Properties of **path length**:

- maximum path length grows in the order of n
- average path length grows in the order of $log(n)$

Equivalence of iTrees with Binary Search Trees:

- both are proper binary trees
- a path search ending in an external node is equivalent with an unsuccessful search in a BST
- average path length of an iTree = average path length of unsuccessful search in BST
- $c(n) = 2H(n-1) - 2(\frac{n-1}{n})$
  - n - number of instances
  - $H(i)$ (harmonic number) $= ln(i) + 0.5772156649$ (Euler's constant)

Isolation forest

- IF simply builds an ensemble of independent iTrees

Isolation forest

- IF simply builds an ensemble of independent iTrees
- each iTree is built from a different sub-sample of the original data (sampling is conducted without replacement)

Isolation forest

- IF simply builds an ensemble of independent iTrees
- each iTree is built from a different sub-sample of the original data (sampling is conducted without replacement)
- each iTree will be able to identify different types of anomalies (due to sub-sampling and random partitioning)

Isolation forest

- IF simply builds an ensemble of independent iTrees
- each iTree is built from a different sub-sample of the original data (sampling is conducted without replacement)
- each iTree will be able to identify different types of anomalies (due to sub-sampling and random partitioning)
- IF only needs 2 parameters:
    - number of trees to build - t

Isolation forest

- IF simply builds an ensemble of independent iTrees
- each iTree is built from a different sub-sample of the original data (sampling is conducted without replacement)
- each iTree will be able to identify different types of anomalies (due to sub-sampling and random partitioning)
- IF only needs 2 parameters:
  - number of trees to build - t
  - sub-sampling size ψ

IForest final anomaly score

- x - an instance

IForest final anomaly score

- x - an instance
- $c(n)$ - average of $h(x)$ given the number of instances n
- we use $c(n)$ to normalize $h(x)$

IForest final anomaly score

- x - an instance
- $c(n)$ - average of $h(x)$ given the number of instances n
- we use $c(n)$ to normalize $h(x)$
- final anomaly score becomes:

$$s(x, n) = 2^{-\frac{E(h(x))}{c(n)}}$$

  - $E(h(x))$ - average of $h(x)$ from all the iTrees of the IForest
  - $0 < s \leq 1$

IForest anomaly score interpretation

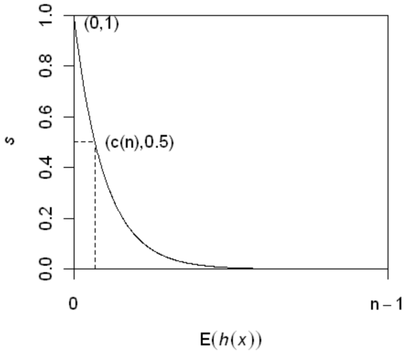


Figure: Relationship of expected path length and anomaly score (Liu, Ting, and Zhou 2008)

IForest anomaly score interpretation

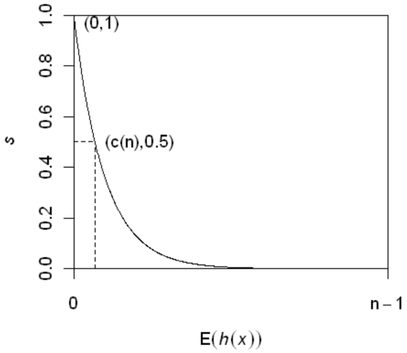


Figure: Relationship of expected path length and anomaly score (Liu, Ting, and Zhou 2008)

- $E(h(x)) \to c(n) \implies s \to 0.5$

IForest anomaly score interpretation

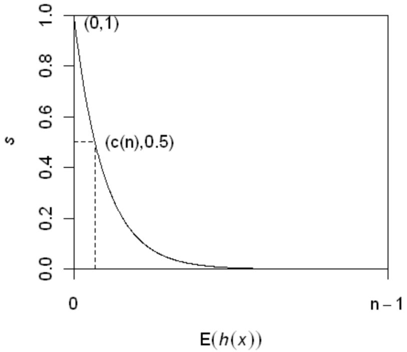


Figure: Relationship of expected path length and anomaly score (Liu, Ting, and Zhou 2008)

- $E(h(x)) \to c(n) \implies s \to 0.5$
- $E(h(x)) \to 0 \implies s \to 1$ (anomaly)

IForest anomaly score interpretation

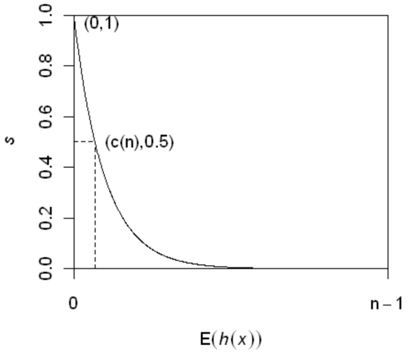


Figure: Relationship of expected path length and anomaly score (Liu, Ting, and Zhou 2008)

- $E(h(x)) \rightarrow c(n) \implies s \rightarrow 0.5$
- $E(h(x)) \rightarrow 0 \implies s \rightarrow 1$ (anomaly)
- $E(h(x)) \rightarrow n-1 \implies s \rightarrow 0$ (normal instances)

IForest anomaly score interpretation

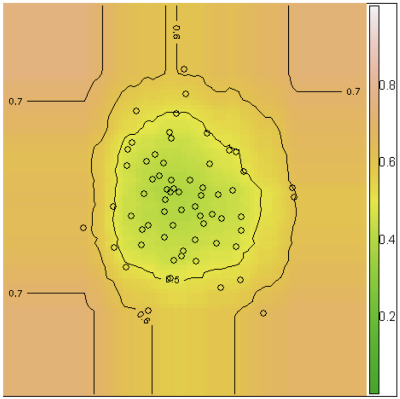


Figure: Anomaly score contour for a Gaussian distribution (Liu, Ting, and Zhou 2008)

IForest anomaly score interpretation

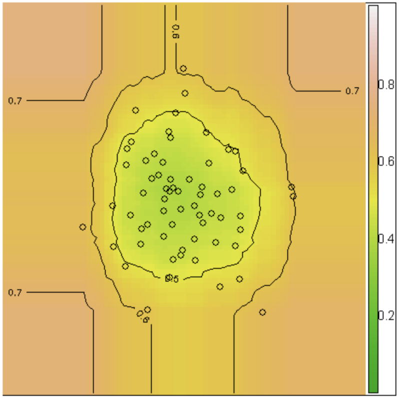


Figure: Anomaly score contour for a Gaussian distribution (Liu, Ting, and Zhou 2008)

- contour lines for different scores
- anomalies can be identified using a threshold value for score s

Masking and Swamping

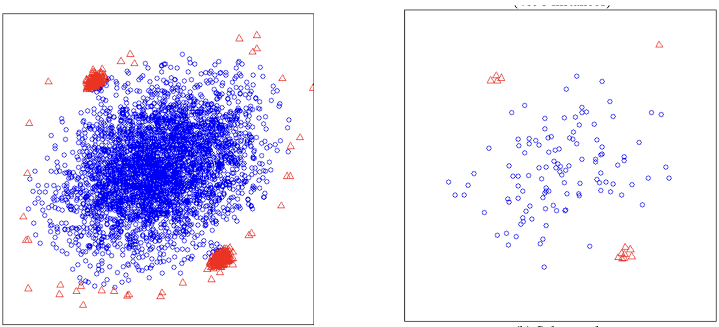
- Masking - the number of anomalies is too large, not allowing the model to detect them

Masking and Swamping

- Masking - the number of anomalies is too large, not allowing the model to detect them
- Swamping - normal instances are wrongly identified as anomalies because they are too close to anomalies

Masking and Swamping

- Masking - the number of anomalies is too large, not allowing the model to detect them
- Swamping - normal instances are wrongly identified as anomalies because they are too close to anomalies



(a) Original sample
(4096 instances)

(b) Sub-sample
(128 instances)

Figure: Effects of masking and swamping (Liu, Ting, and Zhou 2008)

IForest properties

- can only build partial iTrees since we are not interested to isolate every normal point

IForest properties

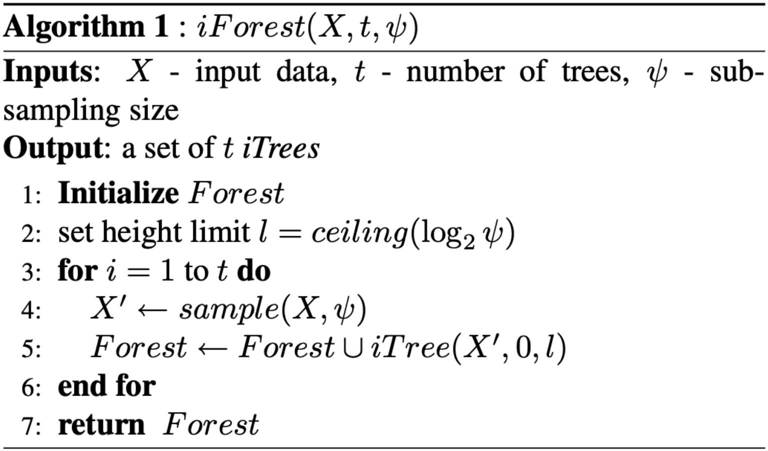
- can only build partial iTrees since we are not interested to isolate every normal point
- no distance or density measure is needed, resulting in a small computational cost

IForest properties

- can only build partial iTrees since we are not interested to isolate every normal point
- no distance or density measure is needed, resulting in a small computational cost
- linear time complexity and low memory requirement
- can handle datasets with large size and dimensionality

IForest training

iTrees are built with recursive partitioning until all instances are isolated or the height limit is reached ($l = ceiling(\log_2 \psi)$)



Algorithm 1 : $iForest(X, t, \psi)$

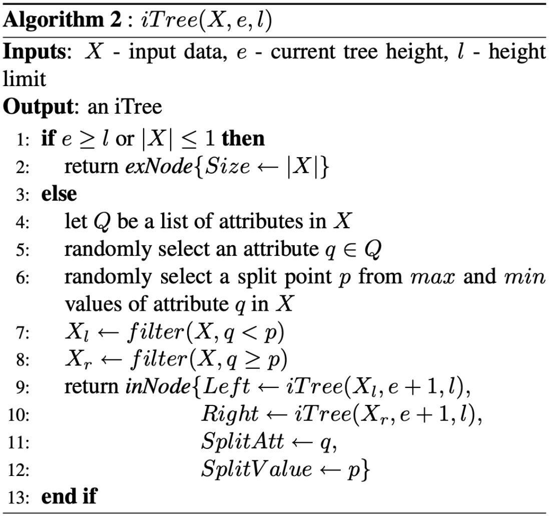
Inputs: X - input data, t - number of trees, ψ - sub-sampling size

Output: a set of t *iTrees*

```
1: Initialize $Forest$
2: set height limit $l = ceiling(\log_2 \psi)$
3: for $i = 1$ to $t$ do
4:     $X' \leftarrow sample(X, \psi)$
5:     $Forest \leftarrow Forest \cup iTree(X', 0, l)$
6: end for
7: return $Forest$
```

Figure: Training algorithm (Liu, Ting, and Zhou 2008)

IForest training



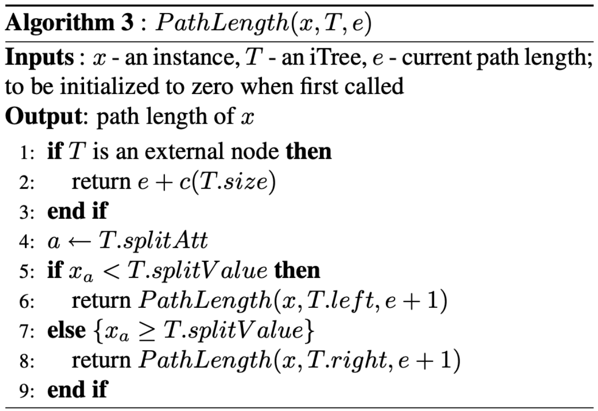
```
Algorithm 2 : $iTree(X, e, l)$
Inputs: $X$ - input data, $e$ - current tree height, $l$ - height limit
Output: an iTree
 1: if $e \geq l$ or $|X| \leq 1$ then
 2:     return exNode$\{Size \leftarrow |X|\}$
 3: else
 4:     let $Q$ be a list of attributes in $X$
 5:     randomly select an attribute $q \in Q$
 6:     randomly select a split point $p$ from $max$ and $min$
        values of attribute $q$ in $X$
 7:     $X_l \leftarrow filter(X, q < p)$
 8:     $X_r \leftarrow filter(X, q \geq p)$
 9:     return inNode$\{Left \leftarrow iTree(X_l, e+1, l),$
10:                  $Right \leftarrow iTree(X_r, e+1, l),$
11:                  $SplitAtt \leftarrow q,$
12:                  $SplitValue \leftarrow p\}$
13: end if
```

Figure: Training algorithm for each iTree (Liu, Ting, and Zhou 2008)

IForest training details

- sub-sampling size $\psi = 256$ (height limit $l = 8$)
- the ensemble size $t = 100$
- training complexity: $O(t\psi \log \psi)$

IForest inference details



Algorithm 3 : $PathLength(x, T, e)$

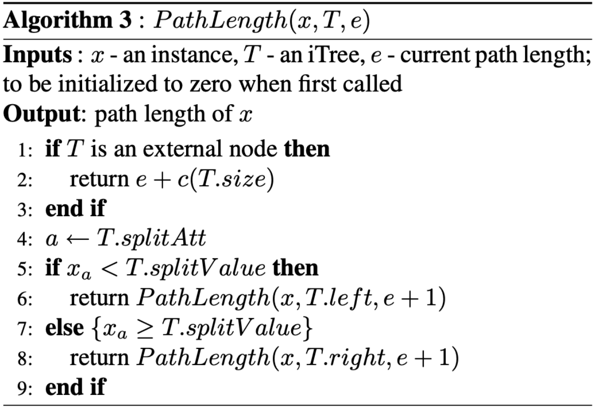
Inputs : x - an instance, T - an iTree, e - current path length; to be initialized to zero when first called

Output: path length of x

```
1: if $T$ is an external node then
2:     return $e + c(T.size)$
3: end if
4: $a \leftarrow T.splitAtt$
5: if $x_a < T.splitValue$ then
6:     return $PathLength(x, T.left, e + 1)$
7: else $\{x_a \geq T.splitValue\}$
8:     return $PathLength(x, T.right, e + 1)$
9: end if
```

Figure: PathLength algorithm (Liu, Ting, and Zhou 2008)

IForest inference details



Algorithm 3 : $PathLength(x, T, e)$

Inputs : x - an instance, T - an iTree, e - current path length; to be initialized to zero when first called

Output: path length of x

```
1: **if** $T$ is an external node **then**
2:     return $e + c(T.size)$
3: **end if**
4: $a \leftarrow T.splitAtt$
5: **if** $x_a < T.splitValue$ **then**
6:     return $PathLength(x, T.left, e + 1)$
7: **else** $\{x_a \geq T.splitValue\}$
8:     return $PathLength(x, T.right, e + 1)$
9: **end if**
```

Figure: PathLength algorithm (Liu, Ting, and Zhou 2008)

- inference complexity: $O(nt \log \psi)$

IForest limitations

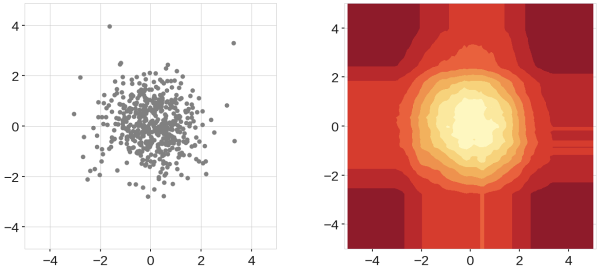


Figure: Data and anomaly score map for IF (Hariri, Kind, and Brunner 2019)

- 2-dimensional normal distributed points
- darker areas $\rightarrow$ higher anomaly scores

IForest limitations

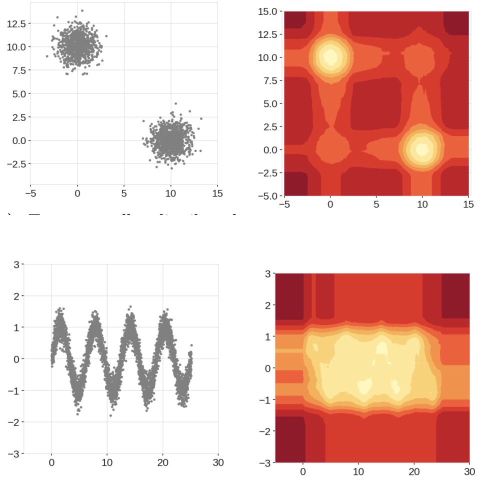


Figure: Data and anomaly score maps for IF (Hariri, Kind, and Brunner 2019)

IForest limitations

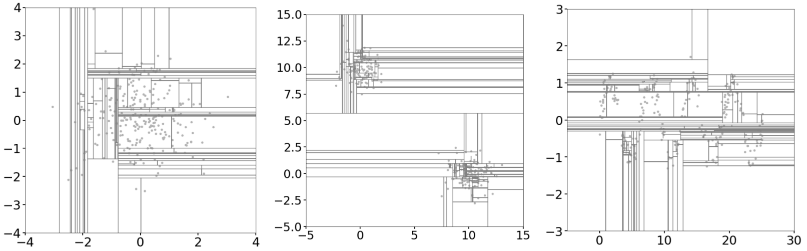


Figure: Branch cuts generated by IF (Hariri, Kind, and Brunner 2019)

IForest limitations

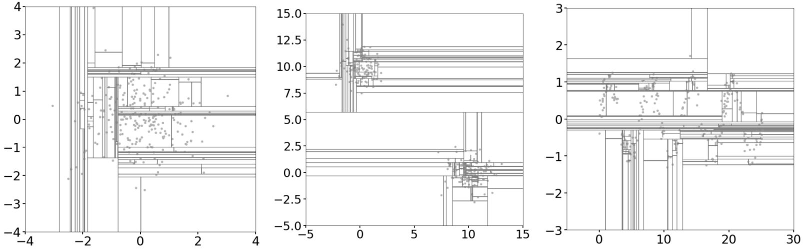


Figure: Branch cuts generated by IF (Hariri, Kind, and Brunner 2019)

- brach cuts tend to cluster where most of the data points reside
- the fact that separating hyperplanes can only be horizontal or vertical causes the presence of many separating hyperplanes in regions with low density

Rotated trees

- every individual tree is built using standard IF

Rotated trees

- every individual tree is built using standard IF
- before that, the sub-sample used for each tree is rotated by a random angle

Rotated trees

- every individual tree is built using standard IF
- before that, the sub-sample used for each tree is rotated by a random angle
- in testing phase, before computing **PathLength**, we have to rotate the test data by the angle corresponding to each tree

Rotated trees

- every individual tree is built using standard IF
- before that, the sub-sample used for each tree is rotated by a random angle
- in testing phase, before computing **PathLength**, we have to rotate the test data by the angle corresponding to each tree
- the bias (corresponding to standard IF) still exists, but now it's different for each tree

Extended Isolation Forest

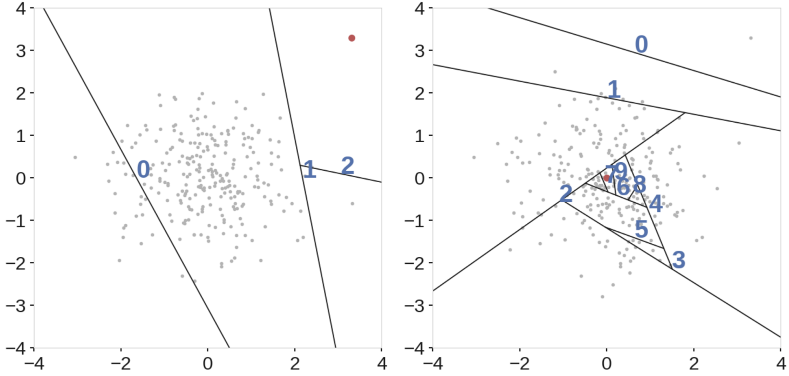


Figure: EIF partitions for anomaly and normal point (Hariri, Kind, and Brunner 2019)

Extended Isolation Forest

Instead of a random attribute and a random value for it we now need:

- a random slope for the separating hyperplane (normal vector $\vec{n}$ to the hyperplane)
- a random intercept $\vec{p}$ for the hyperplane

Extended Isolation Forest

Instead of a random attribute and a random value for it we now need:

- a random slope for the separating hyperplane (normal vector $\vec{n}$ to the hyperplane)
- a random intercept $\vec{p}$ for the hyperplane

Partitioning criteria now becomes:

$$(\vec{x} - \vec{p}) * \vec{n} \leq 0$$

If the condition holds $\vec{x}$ goes to the left branch, else to the right

Extended Isolation Forest

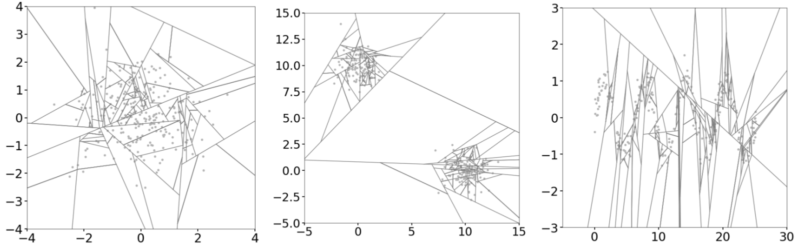


Figure: Branch cuts generated by EIF (Hariri, Kind, and Brunner 2019)

Regions with higher density contain most of the branch cuts while **regions with low density don't end up with many separating hyperplanes (as in standard IF case)**

Extended Isolation Forest - Score maps

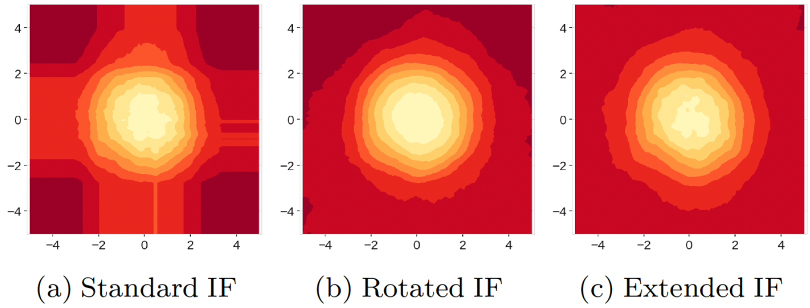


Figure: Anomaly score maps for the single blob, normally distributed (Hariri, Kind, and Brunner 2019)

Extended Isolation Forest - Score maps

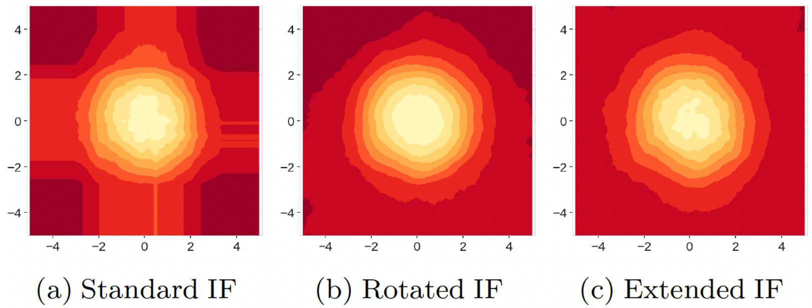


Figure: Anomaly score maps for the single blob, normally distributed (Hariri, Kind, and Brunner 2019)

- low-score bands (present in IF) disappear in EIF

Extended Isolation Forest - Score maps

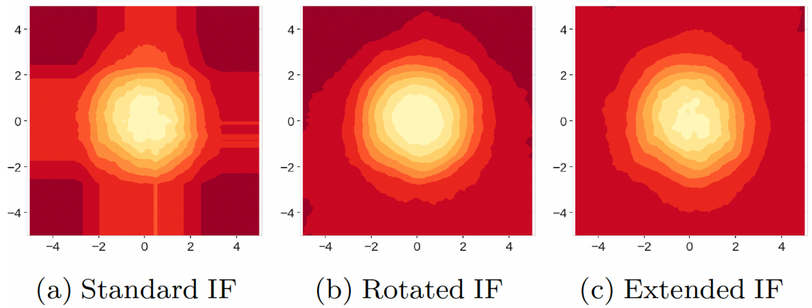
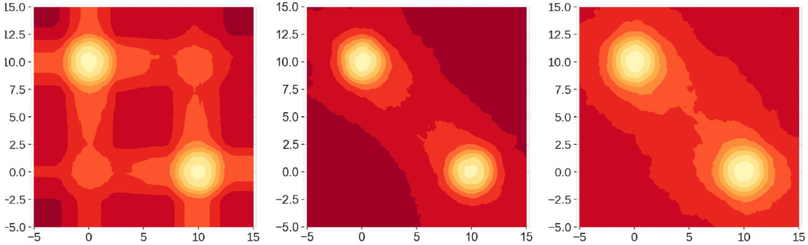


Figure: Anomaly score maps for the single blob, normally distributed (Hariri, Kind, and Brunner 2019)

- low-score bands (present in IF) disappear in EIF
- in EIF score increases monotonically in every direction as we move away from data mean

Extended Isolation Forest - Score maps



(a) Standard IF (b) Rotated IF (c) Extended IF

Figure: Anomaly score maps for the two blobs, normally distributed (Hariri, Kind, and Brunner 2019)

Extended Isolation Forest - Score maps

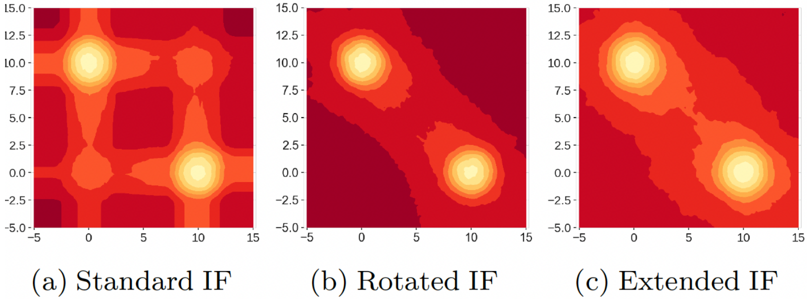


Figure: Anomaly score maps for the two blobs, normally distributed (Hariri, Kind, and Brunner 2019)

- "ghost" regions from the IF score map disappear

Extended Isolation Forest - Score maps

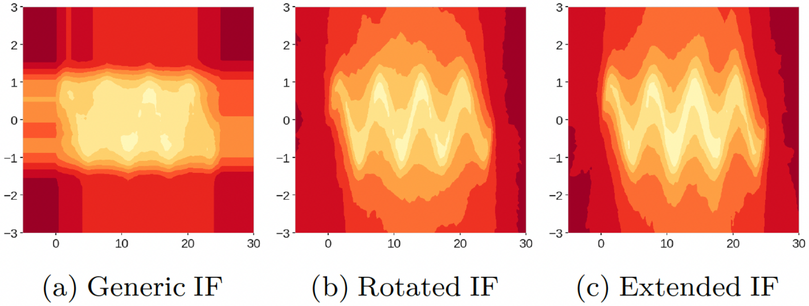


Figure: Anomaly score maps for the sinusoidal data (Hariri, Kind, and Brunner 2019)

Extended Isolation Forest - Score maps

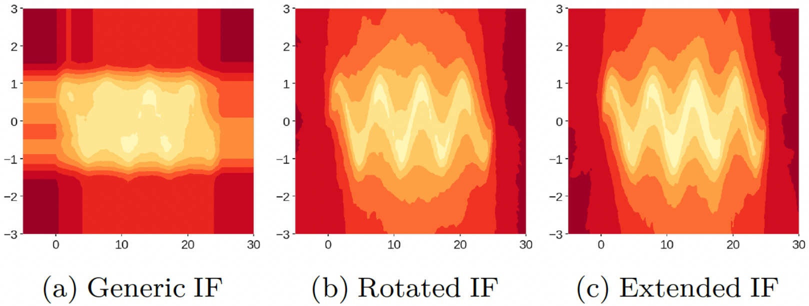


Figure: Anomaly score maps for the sinusoidal data (Hariri, Kind, and Brunner 2019)

- both Rotated IF and EIF capture the structure of the data

Extended Isolation Forest - Score variance

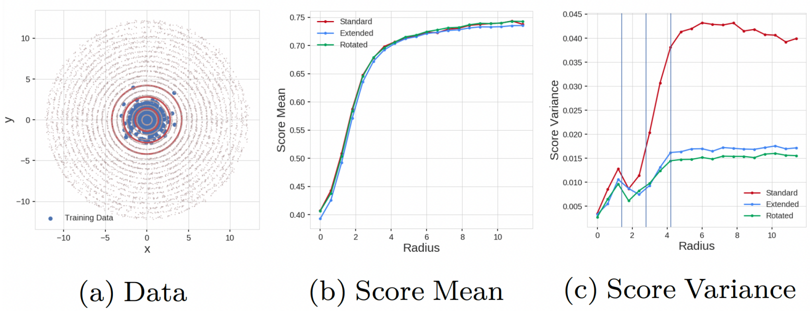


Figure: Anomaly score mean and variance for points located on concentric circles (Hariri, Kind, and Brunner 2019)

Extended Isolation Forest - Score variance

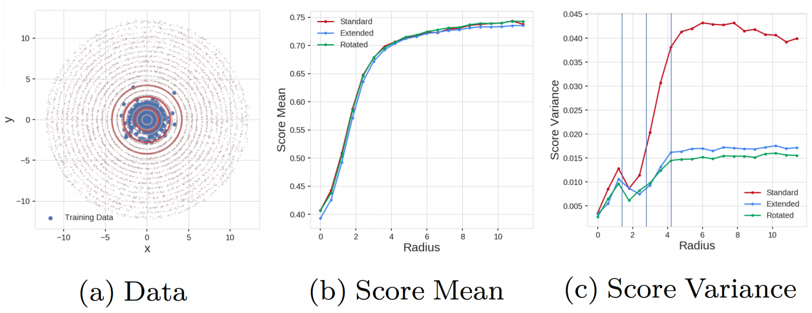


Figure: Anomaly score mean and variance for points located on concentric circles (Hariri, Kind, and Brunner 2019)

- mean varies the same way

Extended Isolation Forest - Score variance

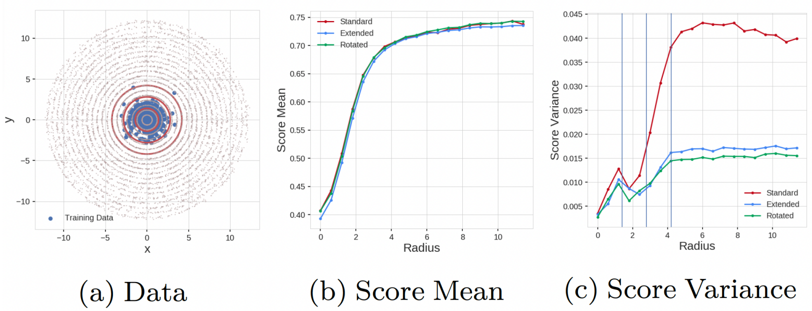
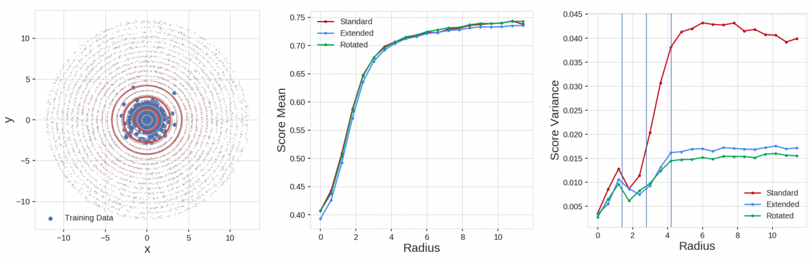


Figure: Anomaly score mean and variance for points located on concentric circles (Hariri, Kind, and Brunner 2019)

- mean varies the same way
- variance is small in all 3 cases for points near the data mean

Extended Isolation Forest - Score variance

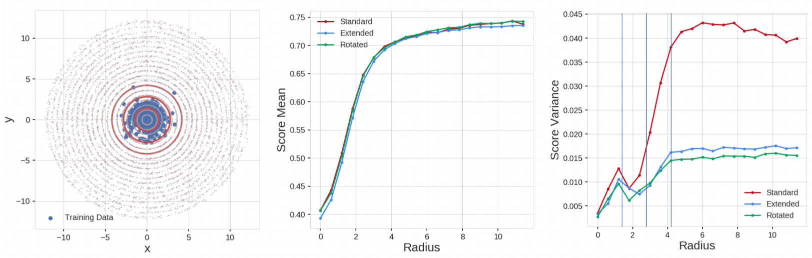


(a) Data (b) Score Mean (c) Score Variance

Figure: Anomaly score mean and variance for points located on concentric circles (Hariri, Kind, and Brunner 2019)

- mean varies the same way
- variance is small in all 3 cases for points near the data mean
- after 3 σ the variance is high for standard IF while for Rotated IF and EIF it stays low

Extended Isolation Forest - Score variance



(a) Data (b) Score Mean (c) Score Variance

Figure: Anomaly score mean and variance for points located on concentric circles (Hariri, Kind, and Brunner 2019)

- mean varies the same way
- variance is small in all 3 cases for points near the data mean
- after 3 σ the variance is high for standard IF while for Rotated IF and EIF it stays low
- **EIF and Rotated IF are more robust!**

EIF limitations

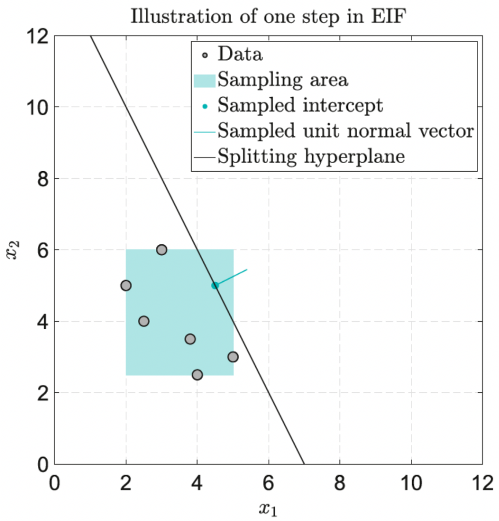


Figure: EIF drawback - generation of empty branches (Lesouple et al. 2021)

EIF limitations

- as dimensionality increases, the probability of selecting an intercept $\vec{p}$ that results in an empty branch increases

EIF limitations

- as dimensionality increases, the probability of selecting an intercept $\vec{p}$ that results in an empty branch increases
- an empty branch wastes time in the process of creation of the trees (it does not separate the dataset in 2 groups)

EIF limitations

- as dimensionality increases, the probability of selecting an intercept $\vec{p}$ that results in an empty branch increases
- an empty branch wastes time in the process of creation of the trees (it does not separate the dataset in 2 groups)

How can we sample $\vec{p}$ in a way that ensures that empty branches can't be created?

Generalized Isolation Forest

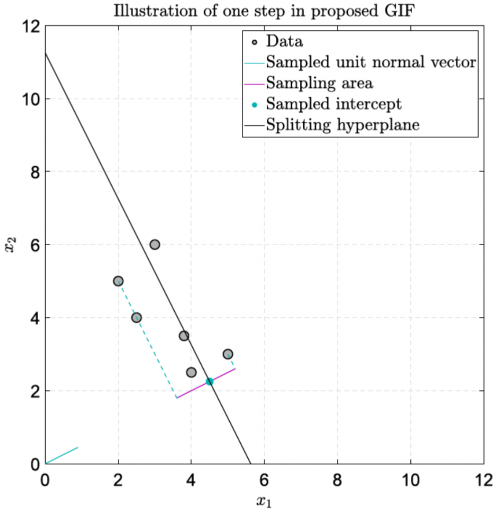


Figure: GIF - intercept sampling (Lesouple et al. 2021)

Generalized Isolation Forest

- normal vector $\vec{n}$ is sampled as in EIF

Generalized Isolation Forest

- normal vector $\vec{n}$ is sampled as in EIF
- we project all data that corresponds to a node to the normal unit vector $\vec{n^{'}}$

$$X^{'} = Xn^{'}$$

Generalized Isolation Forest

- normal vector $\vec{n}$ is sampled as in EIF
- we project all data that corresponds to a node to the normal unit vector $\vec{n'}$

$$X^{'} = Xn^{'}$$

- sample scalar p from minimum and maximum of the projections (uniform distribution)

$$p_{min} = min(X^{'})$$

$$p_{max} = max(X^{'})$$

$$p \sim U([p_{min}; p_{max}])$$

Generalized Isolation Forest

- change branching condition from:

$$(\vec{x} - \vec{p}) * \vec{n} \leq 0$$

Generalized Isolation Forest

- change branching condition from:

$$(\vec{x} - \vec{p}) * \vec{n} \leq 0$$

- to:

$$\vec{x} * \vec{n} \leq p$$

Generalized Isolation Forest

- change branching condition from:

$$(\vec{x} - \vec{p}) * \vec{n} \leq 0$$

- to:

$$\vec{x} * \vec{n} \leq p$$

- GIF obtains very similar performance with EIF (accuracy, AUC and other metrics)

Generalized Isolation Forest

- change branching condition from:

$$(\vec{x} - \vec{p}) * \vec{n} \leq 0$$

- to:

$$\vec{x} * \vec{n} \leq p$$

- GIF obtains very similar performance with EIF (accuracy, AUC and other metrics)
- GIF obtains smaller time than EIF in the tree building process

Deep Isolation Forest

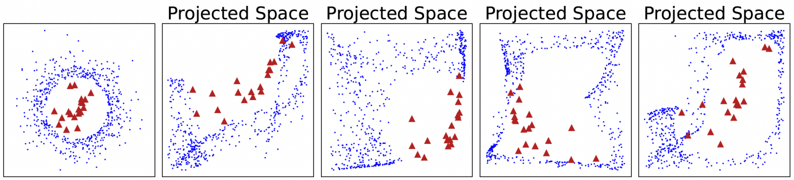


Figure: Hard anomalies in original space and projections obtained with NN (Lesouple et al. 2021)

- anomalies can't be isolated with axis-parallel hyperplanes in original space

Deep Isolation Forest

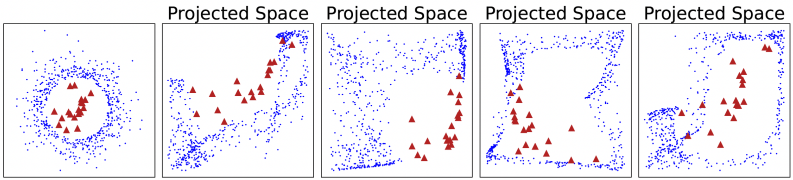


Figure: Hard anomalies in original space and projections obtained with NN (Lesouple et al. 2021)

- anomalies can't be isolated with axis-parallel hyperplanes in original space
- it's possible that they can be isolated using axis-parallel hyperplanes in spaces resulted from a forward pass through randomly initialized NN

Deep Isolation Forest

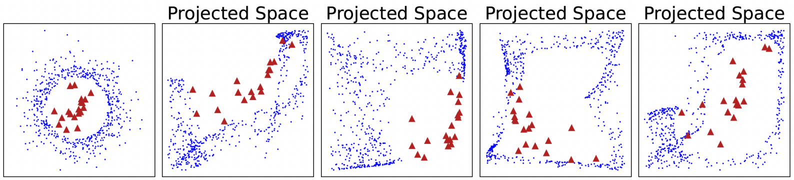


Figure: Hard anomalies in original space and projections obtained with NN (Lesouple et al. 2021)

- anomalies can't be isolated with axis-parallel hyperplanes in original space
- it's possible that they can be isolated using axis-parallel hyperplanes in spaces resulted from a forward pass through randomly initialized NN
- axis parallel cuts in new spaces correspond to nonlinear cuts in the original space

Deep Isolation Forest

- each iTree will have a corresponding neural network with randomly initialized weights

Deep Isolation Forest

- each iTree will have a corresponding neural network with randomly initialized weights
- the random representations ensemble is defined as:

$$\mathcal{G}(\mathcal{D}) = \{\mathcal{X}_u \subset \mathcal{R}^d | \mathcal{X}_u = \Phi_u(\mathcal{D}; \theta_u)\}_{u=1}^{r}$$

r - ensemble size
$\Phi_u : \mathcal{D} \rightarrow \mathcal{R}^d$ - network that maps original data in d-dimensional spaces
θ_u - randomly initialized network weights

Deep Isolation Forest

- each iTree will have a corresponding neural network with randomly initialized weights
- the random representations ensemble is defined as:

$$\mathcal{G}(\mathcal{D}) = \{\mathcal{X}_u \subset \mathcal{R}^d | \mathcal{X}_u = \Phi_u(\mathcal{D}; \theta_u)\}_{u=1}^{r}$$

  r - ensemble size
  $\Phi_u : \mathcal{D} \to \mathcal{R}^d$ - network that maps original data in d-dimensional spaces
  θ_u - randomly initialized network weights
- **DIF does not involve NN optimization**