Extended Isolation Forest

Introduction

The Extended Isolation Forest algorithm generalizes its predecessor algorithm, Isolation Forest. The original Isolation Forest algorithm brings a brand new form of detection, although the algorithm suffers from bias due to tree branching. Extension of the algorithm mitigates the bias by adjusting the branching, and the original algorithm becomes just a special case.

The cause of the bias is that branching is defined by the similarity to BST. At each branching point the feature and the value are chosen; this introduces the bias since the branching point is parallel to one of the axes. The general case needs to define a random slope for each branching point. Instead of selecting the feature and value, it selects a random slope n for the branching cut and a random intercept p. The slope can be generated from $\mathcal{N}(0, 1)$ Gaussian distribution, and the intercept is generated from the uniform distribution with bounds coming from the sub-sample of data to be split. The branching criteria for the data splitting for a given point x is as follows:

$$(x - p) * n \leq 0$$

MOJO Support

Extended Isolation Forest supports importing and exporting MOJOs.

Tutorials and Blogs

The following tutorials are available that describe how to use Extended Isolation Forest:

- Master's thesis: Anomaly detection using Extended Isolation Forest: The thesis deals with anomaly detection algorithms with a focus on the Extended Isolation Forest algorithm and includes the implementation to the

H2O-3 open-source Machine Learning platform.

- Extended Isolation Forest jupyter notebook created by the authors of the algorithm: Describes how Extended Isolation Forest behaves compared to Isolation Forest.

Defining an Extended Isolation Forest Model

Parameters are optional unless specified as *required*.

Algorithm-specific parameters

- extension_level: The number in range $[0, P-1]$; where P is the number of features. The minimum value of the hyperparameter is `0` (default), which corresponds to Isolation Forest behavior. The maximum is $P-1$ and stands for a full extension. As the `extension_level` is increased, the bias of standard Isolation Forest is reduced.
- sample_size: The number of randomly sampled observations used to train each Extended Isolation Forest tree. This option defaults to `256`.
- **disable_training_metrics**: Disable calculating training metrics (expensive on large datasets). This option defaults to `True` (enabled).

Shared tree-algorithm parameters

- ntrees: Specify the number of trees. This option defaults to `100`.
- score_tree_interval: Score the model after every so many trees. This value is set to 0 (disabled) by default.

Common parameters

- categorical_encoding: In case of Extended Isolation Forest, only the ordinal nature of encoding is used for splitting. Specify one of the following encoding schemes for handling categorical features:

    - `auto` or `AUTO` (default): Allow the algorithm to decide. In Isolation Forest, the algorithm will automatically perform `enum` encoding.
    - `enum` or `Enum`: 1 column per categorical feature.
    - `enum_limited` or `EnumLimited`: Automatically reduce categorical levels to the most prevalent ones during training and only keep the **T** (10) most frequent levels.
    - `one_hot_explicit` or `OneHotExplicit`: N+1 new columns for categorical features with N levels.
    - `binary` or `Binary`: No more than 32 columns per categorical feature.
    - `eigen` or `Eigen`: *k* columns per categorical feature, keeping projections of one-hot-encoded matrix onto *k*-dim eigen space only.
    - `label_encoder` or `LabelEncoder`: Convert every enum into the integer of its index (for example, level 0 -> 0, level 1 -> 1, etc.).

- ignore_const_cols: Specify whether to ignore constant training columns since no information can be gained from them. This option defaults to `True` (enabled).
- ignored_columns: (Python and Flow only) Specify the column or columns to be excluded from the model. In Flow, click the checkbox next to a column name to add it to the list of columns excluded from the model. To add all columns, click the **All** button. To remove a column from the list of ignored columns, click the X next to the column name. To remove all columns from the list of ignored columns, click the **None** button. To search for a specific column, type the column name in the **Search** field above the column list. To only show columns with a specific percentage of missing values, specify the percentage in the **Only show columns with more than 0% missing values** field. To change the selections for the hidden columns, use the **Select Visible** or **Deselect Visible** buttons.
- model_id: Specify a custom name for the model to use as a reference. By default, H2O automatically generates a destination key.
- score_each_iteration: (Optional) Enable this option to score during each iteration of the model training (disabled by default).
- seed: Specify the random number generator (RNG) seed for algorithm components dependent on randomization. The seed is consistent for each H2O instance so that you can create models with the same starting conditions in alternative configurations. This option defaults to `-1` (time-based random number).

- training_frame: *Required* Specify the dataset used to build the model.

  NOTE: In Flow, if you click the **Build a model** button from the `Parse` cell, the training frame is entered automatically.

- x: A vector containing the character names of the predictors in the model.

Anomaly Score

The output of Extended Isolation Forest's algorithm is in compliance with this Extended Isolation Forest paper. In short, the anomaly score is the average **mean_length** in a forest normalized by the average path of an unsuccessful search in a binary search tree (BST).

The **anomaly_score**:

$$anomaly\_score(x, sample\_size) = 2^{-mean\_length(x)/c(sample\_size)}$$

where:

$$c(i) = \begin{cases} 2H(i-1) - \frac{2(i-1)}{i} & \text{for } i > 2 \\ 1 & \text{for } i = 2 \\ 0 & \text{otherwise} \end{cases}$$

is the average path of the unsuccessful search in a BST for the data set of size i.

$H(.)$ is a harmonic number estimated as: $H(.) = ln(.) + 0.5772156649$ (Euler's constant)

The **mean_length(x)** is the mean path length of a point in the forest:

$$mean\_length(x) = \frac{path\_length(x) + c(Node.num\_rows)}{ntrees}$$

In case the point x is not isolated, Formula $c(i)$ is used to estimate the tree height from the number of rows in the node. This is done especially for dense clusters of normal points.

The anomaly score is interpreted as follows:

- if instances return an `anomaly_score` very close to 1, then they are definitely anomalies,
- if instances have an `anomaly_score` much smaller than 0.5, then they can be quite safely regarded as normal instances,
- and if all the instances return an `anomaly_score` around 0.5, then the entire sample does not have any distinct anomalies.

Examples

Below is a simple example showing how to build an Extended Isolation Forest model.

R | **Python**

```
import h2o
from h2o.estimators import H2OExtendedIsolationForestEstimator
h2o.init()

# Import the prostate dataset
h2o_df =
h2o.import_file("https://raw.github.com/h2oai/h2o/master/smalldata/logreg/

# Set the predictors
predictors = ["AGE","RACE","DPROS","DCAPS","PSA","VOL","GLEASON"]

# Define an Extended Isolation forest model
eif = H2OExtendedIsolationForestEstimator(model_id = "eif.hex",
                                          ntrees = 100,
                                          sample_size = 256,
                                          extension_level =
len(predictors) - 1)

# Train Extended Isolation Forest
eif.train(x = predictors,
          training_frame = h2o_df)

# Calculate score
eif_result = eif.predict(h2o_df)

# Number in [0, 1] explicitly defined in Equation (1) from Extended
Isolation Forest paper
# or in paragraph '2 Isolation and Isolation Trees' of Isolation Forest
paper
anomaly_score = eif_result["anomaly_score"]

# Average path length  of the point in Isolation Trees from root to the
leaf
mean_length = eif_result["mean_length"]
```

FAQ

- **How does the algorithm handle missing values during training?**

  Extended Isolation Forest cannot handle missing values. You have to put 0 in for those values.

References