Anomaly Detection Service for Transaction and User Activity

Introduction

In the realm of anomaly detection, which plays a crucial role in diverse domains such as fraud detection, cybersecurity, and network monitoring, this project aims to develop a robust machine learning model. The primary objective is to identify potentially anomalous transactions or user activities within a given dataset. The model's capabilities should extend beyond mere anomaly detection by providing insights into the reasons behind the anomalies and identifying the attributes responsible for them. This comprehensive report outlines the approach employed, details the model architecture, and presents the results achieved in the realm of anomaly detection.

Dataset Description

The dataset provided for this project contains the following columns:

Login Timestamp: The timestamp when the login occurred.

User ID: Unique identifier for each user.

IP Address: The IP address associated with the login.

Country: The country from which the login originated.

Region: The region or state from which the login originated.

City: The city from which the login originated.

Browser Name and Version: The name and version of the browser used for the login.

Device Type: The type of device used for the login.

Login Successful: A binary indicator (Yes or No) indicating whether the login was successful or not.

The dataset will be split into a training set and a testing set to evaluate the performance of the anomaly detection model.

Approach and Model Architecture

The approach used for anomaly detection in this project involves a combination of

unsupervised learning and pattern recognition techniques. The model architecture consists of the following steps:

Data Analysis and Preprocessing

Clean the dataset: Handle missing values, remove duplicates, and perform any necessary data cleansing steps.

• 1.4 crore failed login attempt from the same User ID.

• Missing value: browser name version region country. Number of null values of attribute(s) after cleaning:

Handling Timestamp: By breaking down timestamp column into separate components to extract more meaningful information. This was achieved by converting the timestamp into unique features such as year, month, day, hour, minute, and second.

Converting categorical variables: Transform categorical variables, such as browser name and device type, into numerical representations suitable for the model. This was achieved by:

• Browser Info - Hash Encoding: Helps to anonymize the browser information while preserving its categorical nature for anomaly detection.

• Login Successful - Label Encoding: Converts the binary login success variable into numerical labels, allowing anomaly detection algorithms to understand the significance of successful and unsuccessful logins.

• Device Type - One Hot Encoding: Transforms the device type into binary features, enabling anomaly detection models to capture variations in device types accurately.

• IP Address - Octet Separation: Splits the IP address into separate octets, facilitating anomaly detection by capturing anomalous patterns in individual IP address components.

• Country, Region, City - Hash Encoding: Encodes geographical information

into numerical values, enabling anomaly detection algorithms to understand location-based patterns without disclosing sensitive location details.

• User ID - Frequency Encoding: Encodes user IDs based on their frequency, indicating user behavior, and enabling anomaly detection models to identify unusual user activities based on their relative occurrence.

Analyzing the Data

• Number of successful/unsuccessful login:

• Daily Login Success/Failure rate:

• Login status for every device type:

Feature Engineering

Extracting relevant features from the dataset that can capture patterns or anomalies.

• Dimensionality Reduction: Applied t-SNE and Principal Component Analysis (PCA) to reduce the dimensionality of the dataset while preserving essential information.

• Time-Based Features: breaking down timestamp column into separate components to extract more meaningful information.

• Categorical Encoding: Under Data preprocessing

Anomaly Detection

Model Training:

In the model training phase, multiple unsupervised learning algorithms were applied for anomaly detection on the preprocessed dataset. The purpose was to select the most suitable algorithm based on its effectiveness and efficiency in detecting anomalies.

The algorithms were implemented as follows:

1. Isolation Forest: Isolation Forest is an ensemble-based algorithm that isolates anomalies by randomly partitioning the data and constructing isolation trees. It identifies

outliers as data points that require fewer partitions to isolate. This approach is efficient and effective for detecting anomalies in large datasets.

2. Deep Autoencoding Gaussian Mixture Model (DAGMM): DAGMM combines deep autoencoders and Gaussian Mixture Models (GMM) to learn the underlying data distribution. It leverages the reconstruction error of the autoencoder and the log-likelihood of the GMM to identify anomalies. This approach is particularly suitable for capturing complex data patterns and identifying subtle anomalies.

3. Deep Support Vector Data Description (Deep SVDD): Deep SVDD utilizes deep autoencoders and support vector machines to learn a compact representation of the normal data. It aims to enclose the normal instances within a hypersphere in the latent space while minimizing the volume of the hypersphere. Data points falling outside this hypersphere are considered anomalies.

4. K-means Clustering: K-means partitions the dataset into a predefined number of clusters by minimizing the sum of squared distances between the data points and the centroid of each cluster. The algorithm iteratively updates the centroids until convergence.

5. DBSCAN (Density-Based Spatial Clustering of Applications with Noise): DBSCAN groups together dense regions of data points based on their distance and density. It defines core points, which have enough neighboring points within a specified radius, and expands the clusters by connecting density-reachable points.

6. Spectral Clustering: Spectral Clustering projects the dataset into a lower-dimensional space using the spectral embedding technique and then applies a clustering algorithm. It is effective for datasets with non-linear and complex structures.

7. Gaussian Mixture Models (GMM): GMM assumes that the data points are generated from a mixture of Gaussian distributions. It estimates the parameters of the Gaussian components and assigns each data point to the cluster based on the probability distribution.

After applying these algorithms, it was observed that ensemble models and stacking gave the most accurate results.

API Development

- Developed an API that can receive a single record as input and predict whether it is anomalous.
- Implemented functionality to predict anomalies and patterns for a set of records for a given day.